


```
DDDDDDDD  IIIIII  SSSSSSSS  KK  KK  QQQQQQ  UU  UU  000000  TTTTTTTTTT  AAAAAA
DDDDDDDD  IIIIII  SSSSSSSS  KK  KK  QQQQQQ  UU  UU  000000  TTTTTTTTTT  AAAAAA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DD  DD  SS  SS  SS  SS  KK  KK  KK  KK  QQ  QQ  UU  UU  00  00  TT  TT  AA  AA
DDDDDDDD  IIIIII  SSSSSSSS  KK  KK  QQQQ  QQ  UUUUUUUUU  000000  TT  AA  AA
DDDDDDDD  IIIIII  SSSSSSSS  KK  KK  QQQQ  QQ  UUUUUUUUU  000000  TT  AA  AA
                                     ....
                                     ....
                                     ....
                                     ....

LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LL  LL  SS  SS  SS  SS  KK  SS  SSSSSS  SS
LLLLLLLLLL  IIIIII  SSSSSSSS  SS
LLLLLLLLLL  IIIIII  SSSSSSSS  SS
```

1 9
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46

VAX-11 Bliss-32 V4.0-742
[DISKQ.SRC]DISKQUOTA.B32;1

Page 1
(1)

```
0001 0
0002 0 MODULE DISKQUOTA ( ! Disk quota maintenance utility
0003 0 LANGUAGE (BLISS32),
0004 0 MAIN = DISK QUOTA,
0005 0 ADDRESSING_MODE (EXTERNAL = GENERAL,
0006 0 NONEXTERNAL = LONG_RELATIVE),
0007 0 IDENT = 'V04-000'
0008 0 ) =
0009 1 BEGIN
0010 1
0011 1
0012 1
0013 1
0014 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0015 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0016 1 * ALL RIGHTS RESERVED.
0017 1
0018 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0019 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0020 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0021 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0022 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0023 1 * TRANSFERRED.
0024 1
0025 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0026 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0027 1 * CORPORATION.
0028 1
0029 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0030 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0031 1
0032 1
0033 1
0034 1
0035 1 ++
0036 1
0037 1 FACILITY: VMS System Manager Utilities
0038 1
0039 1 ABSTRACT:
0040 1
0041 1 This program implements the commands necessary to maintain the
0042 1 quota file on a files-11 structure level 2 disk. Functions are
0043 1 provided to create the quota file, enable and disable quotas,
0044 1 add, list, modify, and remove authorization entries.
0045 1
0046 1 ENVIRONMENT:
0047 1
0048 1 VAX/VMS Operating System
0049 1
0050 1 --
0051 1
0052 1
0053 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 19-Jun-1979 18:54
0054 1
0055 1 MODIFIED BY:
0056 1
0057 1 V03-003 LMP0140 L. Mark Pilant, 23-Aug-1983 12:57
```



```
58      0058      1      Add support for alphanumeric UICs.
59      0059      1
60      0060      1      V03-002 LMP0133      L. Mark Pilant,      4-Aug-1983 12:14
61      0061      1      Don't set the protection for the created quota file. Let
62      0062      1      the ACP determine it.
63      0063      1
64      0064      1      V03-001 ACG0288      Andrew C. Goldstein,      16-Apr-1982 9:38
65      0065      1      Add DO_IO entry point for REBUILD
66      0066      1
67      0067      1      V02-006 MLJ0058      Martin L. Jack, 4-Nov-1981 20:16
68      0068      1      Extend PLIT in ACT_CREATE so that newly initialized quota file
69      0069      1      does not contain garbage in last few longwords.
70      0070      1
71      0071      1      V02-005 STJ0055      Steven T. Jeffreys,      29-Jun-1981
72      0072      1      Changed external references to use general addressing mode.
73      0073      1
74      0074      1      V0004      ACG0129      Andrew C. Goldstein,      25-Jan-1980 19:28
75      0075      1      Use common REBUILD routine
76      0076      1
77      0077      1      V0003      ACG0087      Andrew C. Goldstein,
78      0078      1      Steve Jeffreys,      20-Nov-1979 20:41
79      0079      1      Add help facility, remove EXAMINE command, add EXIT command
80      0080      1      Add overdraft limit, default values for ADD
81      0081      1
82      0082      1      V0002      ACG0056      Andrew C. Goldstein,      8-Aug-1979 14:49
83      0083      1      Fix REBUILD function to work on non-volume sets
84      0084      1
85      0085      1      **
86      0086      1
87      0087      1
88      0088      1      LIBRARY 'SYSS$LIBRARY:LIB.L32';
89      0089      1      LIBRARY 'SYSS$LIBRARY:TPAMAC.L32';
90      0090      1
91      0091      1
92      0092      1      FORWARD ROUTINE
93      0093      1      DISK QUOTA,      : main routine
94      0094      1      INV_COMMAND,      : signal invalid command
95      0095      1      INV_SWITCH,      : signal invalid switch
96      0096      1      SAVE_KEY,      : save HELP key descriptor
97      0097      1      USE_DEFAULT      : NOVALUE,      : set up default device
98      0098      1      DEF_HANDLER      : NOVALUE,      : condition handler for above
99      0099      1      ACT_USE,      : USE command
100     0100      1      ACT_CREATE,      : CREATE command
101     0101      1      ACT_ENABLE,      : ENABLE command
102     0102      1      ACT_DISABLE,      : DISABLE command
103     0103      1      ACT_ADD,      : ADD command
104     0104      1      ACT_REMOVE,      : REMOVE command
105     0105      1      ACT_SHOW,      : SHOW command
106     0106      1      ACT_MODIFY,      : MODIFY command
107     0107      1      ACT_REBUILD,      : REBUILD command
108     0108      1      ACT_HELP,      : HELP command
109     0109      1      MAIN_HANDLER,      : facility condition handler
110     0110      1      EXIT_HANDLER      : NOVALUE,      : facility exit handler
111     0111      1      COMMON_IO;      : common I/O routine for DO_IO calls
112     0112      1
113     0113      1      : Structure declarations used for system defined structures to
114     0114      1      : save typing.
```

```
115 0115 1 !
116 0116 1 ! STRUCTURE
117 0117 1 BBLOCK [O, P, S, E; N] =
118 0118 1 [N]
119 0119 1 (BBLOCK+O)<P,S,E>,
120 0120 1
121 0121 1 BBLOCKVECTOR [I, O, P, S, E; N, BS] =
122 0122 1 [N*BS]
123 0123 1 ((BBLOCKVECTOR+I*BS)+O)<P,S,E>,
124 0124 1
125 0125 1 EXIT_CTRL_BLK [I : N] =
126 0126 1 [(4+N)*4]
127 0127 1 (EXIT_CTRL_BLK+I*4)<0,32,0>;
128 0128 1 ! exit handler descriptor
129 0129 1 ! N = # of arguments ( N <= 1 )
130 0130 1 ! the block is a longword array
131 0131 1
132 0132 1 ! Macro to generate a string descriptor.
133 M 0133 1 MACRO
134 0134 1 DESCRIPTOR (STRING) =
135 0135 1 UPLIT (%CHARCOUNT (STRING), UPLIT BYTE (STRING))%;
136 0136 1
137 0137 1 ! Macro to signal error exit.
138 0138 1 MACRO
139 M 0139 1 ERR_EXIT [] =
140 M 0140 1 SIGNAL_STOP (%REMAINING)
141 0141 1 %;
142 0142 1
143 0143 1 ! Macro to signal error message.
144 0144 1 MACRO
145 0145 1 ERR_MESSAGE [] =
146 M 0146 1 SIGNAL (%REMAINING)
147 M 0147 1 %;
148 0148 1
149 0149 1 ! Macro to declare argument list in TPARSE action routine.
150 0150 1 MACRO
151 0151 1 TPARSE_ARGS =
152 0152 1 BUILTIN AP;
153 M 0153 1 BIND TPARSE_BLOCK = AP : REF BBLOCK;
154 M 0154 1 %;
155 M 0155 1
156 0156 1
```

```
158 0157 1 ;+
159 0158 1
160 0159 1 Error messages
161 0160 1
162 0161 1 Macro to generate each error message.
163 0162 1
164 0163 1 ;-
165 0164 1
166 0165 1 MACRO
167 M 0166 1 ERR_TEXT (CODE, COUNT, SEVERITY, STRING) =
168 M 0167 1 LITERAL %NAME ('DSKQ$', CODE) = MSG_CODE + FAC_CODE*16;
169 M 0168 1 SWITCHES UNAMES;
170 M 0169 1 PSECT OWN = $MSG_TEXT;
171 M 0170 1 OWN MSG_TEXT : VECTOR [%CHARCOUNT(CODE)+11+%CHARCOUNT(STRING)+2, BYTE]
172 M 0171 1 INITIAL (BYTE (COUNT,
173 M 0172 1 %CHARCOUNT(CODE)+11+%CHARCOUNT(STRING),
174 M 0173 1 '%DISKQ-$', %STRING (SEVERITY), '- ',
175 M 0174 1 %STRING (CODE), ', ', STRING));
176 M 0175 1
177 M 0176 1 PSECT OWN = $MSG_INDEX;
178 M 0177 1 OWN MSG_INDEX : INITIAL (MSG_TEXT);
179 M 0178 1 UNDECLARE MSG_TEXT, MSG_INDEX;
180 M 0179 1 SWITCHES NOUNAMES;
181 M 0180 1 %ASSIGN (MSG_CODE, MSG_CODE+8)
182 M 0181 1 PSECT OWN = $OWNS;
183 M 0182 1 %;
184 0183 1
185 0184 1 ; Initialize and label the message sections.
186 0185 1
187 0186 1
188 0187 1 PSECT
189 0188 1 OWN = $MSG_TEXT (NOWRITE, ALIGN(0));
190 0189 1 OWN
191 0190 1 MESSAGE_TEXT : VECTOR [0, BYTE];
192 0191 1 PSECT
193 0192 1 OWN = $MSG_INDEX (NOWRITE, ALIGN (2));
194 0193 1 OWN
195 0194 1 MESSAGE_TABLE : VECTOR [0];
196 0195 1
197 0196 1 COMPILETIME
198 0197 1 MSG_CODE = 0;
199 0198 1
200 0199 1 ;
201 0200 1 ; Generate the error messages
202 0201 1 ;
203 0202 1
204 0203 1 LITERAL
205 0204 1 FAC_CODE = 69; ! or whatever
206 0205 1
207 0206 1
208 0207 1 ERR_TEXT (CMD_ERR, 0, F, 'I/O error reading commands');
209 0208 1 ERR_TEXT (INV_CMD, 6, E, 'unrecognized command!/:AD\!AD\!AD');
210 0209 1 ERR_TEXT (AMB_CMD, 6, E, 'ambiguous command!/:AD\!AD\!AD');
211 0210 1 ERR_TEXT (INV_QUAL, 6, E, 'unrecognized qualifier!/:AD\!AD\!AD');
212 0211 1 ERR_TEXT (AMB_QUAL, 6, E, 'ambiguous qualifier!/:AD\!AD\!AD');
213 0212 1 ERR_TEXT (INV_UIC, 6, E, 'invalid UIC!/:AD\!AD\!AD');
214 0213 1 ERR_TEXT (SYNTAX, 6, E, 'command syntax error!/:AD\!AD\!AD');
```


| | | | | | |
|-----|------|---|----------|-------------|--|
| 215 | 0214 | 1 | ERR_TEXT | (NONLOCAL, | 0, E, 'device is not a local device'); |
| 216 | 0215 | 1 | ERR_TEXT | (NOTRAN, | 0, E, 'logical name is recursively defined'); |
| 217 | 0216 | 1 | ERR_TEXT | (NODEVICE, | 0, E, 'no device currently selected'); |
| 218 | 0217 | 1 | ERR_TEXT | (CREATERR, | 0, E, 'error creating quota file'); |
| 219 | 0218 | 1 | ERR_TEXT | (INITERR, | 0, E, 'error initializing quota file'); |
| 220 | 0219 | 1 | ERR_TEXT | (CLOSERR, | 0, E, 'error closing quota file'); |
| 221 | 0220 | 1 | ERR_TEXT | (ACTERR, | 0, E, 'failed to enable quota file'); |
| 222 | 0221 | 1 | ERR_TEXT | (DACTERR, | 0, E, 'failed to disable quota file'); |
| 223 | 0222 | 1 | ERR_TEXT | (ADDERR, | 0, E, 'failed to add quota file entry'); |
| 224 | 0223 | 1 | ERR_TEXT | (REMOVERR, | 0, E, 'failed to remove quota file entry'); |
| 225 | 0224 | 1 | ERR_TEXT | (MODIFYERR, | 0, E, 'failed to modify quota file entry'); |
| 226 | 0225 | 1 | ERR_TEXT | (EXAMINERR, | 0, E, 'cannot examine quota file entry'); |
| 227 | 0226 | 1 | ERR_TEXT | (INUSE, | 3, I, '[!OW,!OW] has !UL blocks in use'); |
| 228 | 0227 | 1 | ERR_TEXT | (LOCKERR, | 0, E, 'failed to lock volume'); |
| 229 | 0228 | 1 | ERR_TEXT | (UNLOCKERR, | 0, E, 'failed to unlock volume'); |
| 230 | 0229 | 1 | ERR_TEXT | (MAXVOLS, | 0, E, 'volume set has too many volumes to handle'); |
| 231 | 0230 | 1 | ERR_TEXT | (ACCINDEXF, | 1, E, 'failed to access index file on relative volume !UW'); |
| 232 | 0231 | 1 | ERR_TEXT | (ACQFILE, | 0, E, 'failed to access quota file'); |
| 233 | 0232 | 1 | ERR_TEXT | (QUOTARERR, | 0, E, 'I/O error reading quota file'); |
| 234 | 0233 | 1 | ERR_TEXT | (BITMAPERR, | 1, E, 'I/O error reading index file bitmap on relative volume !UW'); |
| 235 | 0234 | 1 | ERR_TEXT | (HEADERERR, | 2, W, 'I/O error reading file header !UL on relative volume !UW'); |
| 236 | 0235 | 1 | ERR_TEXT | (MEMALLOC, | 0, E, 'cannot allocate sufficient memory'); |
| 237 | 0236 | 1 | ERR_TEXT | (HOMEBLOCK, | 1, E, 'failed to read home block on relative volume !UW'); |
| 238 | 0237 | 1 | ERR_TEXT | (HELP_INIT, | 1, E, 'failed help library index init'); |
| 239 | 0238 | 1 | ERR_TEXT | (HELP_OPEN, | 1, E, 'failed to open help library'); |
| 240 | 0239 | 1 | ERR_TEXT | (HELP_TEXT, | 1, E, 'failed to access help text'); |

```
242 0240 1 |
243 0241 1 | Module own storage.
244 0242 1 |
245 0243 1 | LITERAL
246 0244 1 |     COMMAND_LENGTH = 132,
247 0245 1 |     OUTPUT_LENGTH  = 132,
248 0246 1 |     MAX_KEYS       = 14,           ! 2*(max # of keys) for HELP commnad
249 0247 1 |
250 0248 1 | The following are indexes into the Exit Handler Control Block
251 0249 1 |
252 0250 1 |     XHNDLR_ADDRESS = 1,           ! exit handler address
253 0251 1 |     XHNDLR_ARGCNT  = 2,           ! exit handler argument count
254 0252 1 |     XHNDLR_STSADDR = 3,           ! system exit status address
255 0253 1 |
256 0254 1 | OWN
257 0255 1 |     CHANNEL        : WORD,         ! channel for disk I/O
258 0256 1 |     IO_STATUS      : VECTOR [4, WORD], ! I/O status block
259 0257 1 |     COMMAND_LINE   : VECTOR [COMMAND_LENGTH, BYTE], ! command line buffer
260 0258 1 |     OUTPUT_LINE    : VECTOR [OUTPUT_LENGTH, BYTE], ! output line buffer
261 0259 1 |
262 0260 1 |     COMMAND_DESC   : VECTOR [2] INITIAL (COMMAND_LENGTH, COMMAND_LINE),
263 0261 1 |                               ! command line descriptor
264 0262 1 |     OUTPUT_DESC    : VECTOR [2] INITIAL (OUTPUT_LENGTH, OUTPUT_LINE),
265 0263 1 |                               ! output line descriptor
266 0264 1 |     EXIT_HNDLR_DESC : EXIT_CTRL_BLK [1],
267 0265 1 |                               ! exit handler descriptor
268 0266 1 |
269 0267 1 |
270 0268 1 | Area to zero before each command.
271 0269 1 |
272 0270 1 |     ZERO_AREA      : VECTOR [0],
273 0271 1 |
274 0272 1 | Cleanup action flags
275 0273 1 |
276 0274 1 |     CLEANUP_FLAGS  : BITVECTOR [32];
277 0275 1 |
278 0276 1 | LITERAL
279 0277 1 |     CLF_UNLOCK     = 0,           ! unlock volume set
280 0278 1 |     CLF_EXIT       = 1,           ! exit command entered
281 0279 1 |
282 0280 1 | Quota file record buffers
283 0281 1 |
284 0282 1 | OWN
285 0283 1 |     SRC_REC        : BBLOCK [DQFSC_LENGTH],
286 0284 1 |     DST_REC        : BBLOCK [DQFSC_LENGTH],
287 0285 1 |
288 0286 1 | FIB for quota file operations
289 0287 1 |
290 0288 1 |     QUOTA_FIB      : BBLOCK [FIBSC_LENGTH],
291 0289 1 |
292 0290 1 | TPARSE action routine output
293 0291 1 |
294 0292 1 |     UIC_FLAGS      : BITVECTOR [32], ! UIC wild card flags
295 0293 1 |
296 0294 1 |
297 0295 1 | Storage used for HELP function.
298 0296 1 |
```



```
299 0297 1 KEY_VECTOR : VECTOR [MAX_KEYS], ! use as a descriptor vector
300 0298 1 KEY_INDEX,
301 0299 1
302 0300 1 ZERO_END : VECTOR [0];
303 0301 1
304 0302 1 LITERAL
305 0303 1 ZERO_LENGTH = ZERO_END - ZERO_AREA;
306 0304 1
307 0305 1 Quota record descriptors
308 0306 1
309 0307 1 OWN
310 0308 1 SRCREC_DESC : VECTOR [2] INITIAL (DQF$C_LENGTH, SRC_REC),
311 0309 1 DSTREC_DESC : VECTOR [2] INITIAL (DQF$C_LENGTH, DST_REC),
312 0310 1 QFIB_DESC : VECTOR [2] INITIAL (FIB$C_LENGTH, QUOTA_FIB);
313 0311 1
314 0312 1 TPARSE interface and output
315 0313 1
316 0314 1 LITERAL
317 0315 1 WILD_GROUP = $BITPOSITION (FIB$V_ALL_GRP),
318 0316 1 WILD_MEMBER = $BITPOSITION (FIB$V_ALL_MEM),
319 0317 1 PERM_SPEC = $BITPOSITION (FIB$V_MOD_PERM),
320 0318 1 OVER_SPEC = $BITPOSITION (FIB$V_MOD_OVER);
321 0319 1
322 0320 1 OWN
323 0321 1 TPARSE_BLOCK : BBLOCK [TPASK_LENGTH0]
324 0322 1 INITIAL (TPASK_COUNT0, TPASM_ABBREV);
325 0323 1
326 0324 1 BIND
327 0325 1 UIC_VALUE = SRC_REC[DQF$L_UIC], ! full UIC
328 0326 1 PERM_VALUE = SRC_REC[DQF$L_PERMQUOTA], ! permanent quota
329 0327 1 OVER_VALUE = SRC_REC[DQF$L_OVERDRAFT]; ! overdraft limit
330 0328 1
331 0329 1 PSECT PLIT = $OWNS;
332 0330 1
333 0331 1 BIND
334 0332 1 QFILE_NAME = DESCRIPTOR ('QUOTA.SYS;1'); ! quota file name
335 0333 1
336 0334 1 PSECT PLIT = $PLIT$;
```

```
338 0335 1 GLOBAL ROUTINE DISK_QUOTA =
339 0336 1
340 0337 1 ++
341 0338 1
342 0339 1 Functional Description:
343 0340 1
344 0341 1 This is the main program of the disk quota utility. It accepts
345 0342 1 commands from SYS$INPUT, parses and processes them, and reports
346 0343 1 errors.
347 0344 1
348 0345 1 Calling Sequence:
349 0346 1 standard
350 0347 1
351 0348 1 Input Parameters:
352 0349 1 none
353 0350 1
354 0351 1 Implicit Inputs:
355 0352 1 none
356 0353 1
357 0354 1 Output Parameters:
358 0355 1 none
359 0356 1
360 0357 1 Implicit Outputs:
361 0358 1 none
362 0359 1
363 0360 1 Routines Called:
364 0361 1 none
365 0362 1
366 0363 1 Routine Value:
367 0364 1 none
368 0365 1
369 0366 1 Signals:
370 0367 1 none
371 0368 1
372 0369 1 Side Effects:
373 0370 1 none
374 0371 1
375 0372 1 --
376 0373 1
377 0374 2 BEGIN
378 0375 2
379 0376 2 LOCAL
380 0377 2 STATUS, ! general status value
381 0378 2 P; ! general string pointer
382 0379 2
383 0380 2
384 0381 2 Generate translation table to convert lower case to upper case.
385 0382 2
386 0383 2 MACRO
387 M 0384 2 UPCASE_ENTRY (DUMMY) [] =
388 M 0385 2 %IF ((%COUNT AND %X'7F') GEQU 'a') AND ((%COUNT AND %X'7F') LEQU 'z')
389 M 0386 2 %THEN (%COUNT AND %X'5F')
390 M 0387 2 %ELSE (%COUNT AND %X'7F')
391 M 0388 2 %FI
392 M 0389 2 %IF %COUNT LSSU 255
393 M 0390 2 %THEN , UPCASE_ENTRY (0)
394 M 0391 2 %FI
```

```

0392      Z;
0393
0394      BIND
0395      UPCASE_TABLE = UPLIT BYTE (UPCASE_ENTRY (0));
0396
0397      EXTERNAL LITERAL
0398      LIB$SYNTAXERR;
0399
0400      EXTERNAL ROUTINE
0401      LIB$GET_INPUT : ADDRESSING_MODE (GENERAL); ! get line from SYS$INPUT
0402      LIB$PARSE : ADDRESSING_MODE (GENERAL); ! parse and process command
0403
0404

```



```
409 0405 2  | TPARSE state table to parse commands.
410 0406  |
411 0407  |
412 0408  |
413 0409  | $INIT_STATE (STATE_TABLE, KEY_TABLE);
414 0410  |
415 0411  |
416 0412  | Initial state - acquire command.
417 0413  |
418 0414  |
419 P 0415  | $STATE (START,
420 P 0416  |      ('ADD', DO_ADD),
421 P 0417  |      ('CREATE', MORE, ACT_CREATE),
422 P 0418  |      ('DISABLE', MORE, ACT_DISABLE),
423 P 0419  |      ('ENABLE', MORE, ACT_ENABLE),
424 P 0420  |      ('EXIT', TPAS_EXIT, T^CLF_EXIT, CLEANUP_FLAGS),
425 P 0421  |      ('HELP', DO_HELP),
426 P 0422  |      ('MODIFY', DO_MODIFY),
427 P 0423  |      ('REBUILD', MORE, ACT_REBUILD),
428 P 0424  |      ('REMOVE', DO_REMOVE),
429 P 0425  |      ('SHOW', DO_SHOW),
430 P 0426  |      ('USE', DO_USE),
431 P 0427  |      (TPAS_SYMBOL, INV_COMMAND),
432 P 0428  |      (TPAS_EOS, TPAS_EXIT)
433 0429  | );
434 0430  |
435 0431  |
436 0432  | USE command
437 0433  |
438 0434  |
439 P 0435  | $STATE (DO_USE,
440 P 0436  |      ((DEV_SPEC), MORE, ACT_USE)
441 0437  | );
442 0438  |
443 0439  |
444 0440  | ADD command
445 0441  |
446 0442  |
447 P 0443  | $STATE (DO_ADD,
448 P 0444  |      ((CMD_SWIT), DO_ADD),
449 P 0445  |      ((UIC), DO_ADD1)
450 0446  | );
451 0447  |
452 P 0448  | $STATE (DO_ADD1,
453 P 0449  |      ((CMD_SWIT), DO_ADD1),
454 P 0450  |      (TPAS_LAMBDA, MORE, ACT_ADD)
455 0451  | );
456 0452  |
457 0453  |
458 0454  | MODIFY command
459 0455  |
460 0456  |
461 P 0457  | $STATE (DO_MODIFY,
462 P 0458  |      ((CMD_SWIT), DO_MODIFY),
463 P 0459  |      ((UIC), DO_MODIFY1)
464 0460  | );
465 0461  |
```

```
466 P 0462 2 $STATE (DO MODIFY1,  
467 P 0463 ((CMD_SWIT), DO MODIFY1),  
468 P 0464 (TPAS_LAMBDA, MORE, ACT_MODIFY)  
469 0465 );  
470 0466  
471 0467  
472 0468  
473 0469  
474 0470  
475 P 0471 $STATE (DO_SHOW,  
476 P 0472 ((UIC), MORE, ACT_SHOW)  
477 0473 );  
478 0474  
479 0475  
480 0476  
481 0477  
482 0478  
483 P 0479 $STATE (DO_REMOVE,  
484 P 0480 ((UIC), MORE, ACT_REMOVE)  
485 0481 );  
486 0482  
487 0483  
488 0484  
489 0485  
490 0486  
491 P 0487 $STATE (MORE,  
492 P 0488 (START),  
493 P 0489 (TPAS_EOS, TPAS_EXIT)  
494 0490 );  
495 0491  
496 0492  
497 0493  
498 0494  
499 0495  
500 P 0496 $STATE (CMD_SWIT,  
501 P 0497 ('/'),  
502 0498 );  
503 0499  
504 P 0500 $STATE (  
505 P 0501 ('PERMQUOTA', DO_PERMQUOTA,, 1*PERM_SPEC, UIC_FLAGS),  
506 P 0502 ('OVERDRAFT', DO_OVERDRAFT,, 1*OVER_SPEC, UIC_FLAGS),  
507 P 0503 (TPAS_SYMBOL,, INV_SWITCH)  
508 0504 );  
509 0505  
510 P 0506 $STATE (DO_PERMQUOTA,  
511 P 0507 ('='),  
512 0508 );  
513 0509  
514 P 0510 $STATE (  
515 P 0511 (TPAS_DECIMAL, TPAS_EXIT,,, PERM_VALUE)  
516 0512 );  
517 0513  
518 P 0514 $STATE (DO_OVERDRAFT,  
519 P 0515 ('='),  
520 0516 );  
521 0517  
522 P 0518 $STATE (,
```

```
523 P 0519 (TPAS_DECIMAL, TPAS_EXIT,,, OVER_VALUE)
524   0520 );
525   0521
526   0522
527   0523
528   0524
529   0525
530   0526
531   0527 $STATE (DEV_SPEC,
532   0528 (TPAS_SYMBOL)
533   0529 );
534   0530
535   0531 $STATE (
536   0532 ('.' TPAS_EXIT),
537   0533 (TPAS_LAMBDA, TPAS_EXIT)
538   0534 );
539   0535
540   0536
541   0537
542   0538
543   0539 $STATE (UIC
544   0540 (TPAS_IDENT, TPAS_EXIT,,, UIC_VALUE)
545   0541 );
546   0542
547   0543
548   0544
549   0545
550   0546
551   0547 $STATE (DO_HELP,
552   0548 (TPAS_STRING, DO_HELP, SAVE_KEY),
553   0549 ((DO_QUALIFIER), DO_HELP, SAVE_KEY),
554   0550 ('.' DO_HELP, SAVE_KEY),
555   0551 ((ELIPSIS), DO_HELP, SAVE_KEY),
556   0552 (TPAS_LAMBDA, MORE, ACT_HELP)
557   0553 );
558   0554
559   0555 $STATE (DO_QUALIFIER,
560   0556 ('/'))
561   0557 );
562   0558
563   0559 $STATE (
564   0560 ('PERMQUOTA', TPAS_EXIT),
565   0561 ('OVERDRAFT', TPAS_EXIT),
566   0562 (TPAS_STRING, TPAS_EXIT)
567   0563 );
568   0564
569   0565 $STATE (ELIPSIS,
570   0566 ('.'))
571   0567 );
572   0568
573   0569 $STATE (
574   0570 ('.'))
575   0571 );
576   0572
577   0573 $STATE (
578   0574 ('.', TPAS_EXIT)
579   0575 );
```



```
581 0576
582 0577
583 0578
584 0579
585 0580
586 0581
587 0582
588 0583
589 0584
590 0585
591 0586
592 0587
593 0588
594 0589
595 0590
596 0591
597 0592
598 0593
599 0594
600 0595
601 0596
602 0597
603 0598
604 0599
605 0600
606 0601
607 0602
608 0603
609 0604
610 0605
611 0606
612 0607
613 0608
614 0609
615 0610
616 0611
617 0612
618 0613
619 0614
620 0615
621 0616
622 0617
623 0618
624 0619
625 0620
626 0621
627 0622
628 0623
629 0624
630 0625
631 0626
632 0627
633 0628
634 0629
635 P 0630
636 P 0631
637 P 0632

:
: Set up a channel to the default disk, if it is defined.
:
ENABLE MAIN_HANDLER;
USE_DEFAULT ();

: Set up the exit handler descriptor and declare the handler.
:
EXIT_HNDLR_DESC[XHNDLR_ADDRESS] = EXIT_HANDLER;
EXIT_HNDLR_DESC[XHNDLR_ARGCNT] = 1;
EXIT_HNDLR_DESC[XHNDLR_STSADDR] = EXIT_HNDLR_DESC[XHNDLR_STSADDR+1];
SDCLEXH (DESBK=EXIT_HNDLR_DESC);

: Acquire a command line, convert to upper case, and parse it. Command
: processing is actually done by parser action routines. If a syntax error
: occurs, output an error message. Errors occurring during the command
: processing are signalled at that time.
:
WHILE 1 DO
BEGIN
COMMAND_DESC[0] = COMMAND_LENGTH;
STATUS = LIB$GET_INPUT (COMMAND_DESC, DESCRIPTOR ('DISKQ>'));
IF NOT .STATUS
THEN
BEGIN
IF .STATUS NEQ RMSS_EOF
THEN ERR_MESSAGE (DSKQ$CMD_ERR, .STATUS);
RETURN 1;
END;
CH$TRANSLATE (UPCASE_TABLE, .COMMAND_DESC[0], .COMMAND_DESC[1], 0,
.COMMAND_DESC[0], .COMMAND_DESC[1]);
P = .COMMAND_DESC[0] + .COMMAND_DESC[1];
UNTIL CH$RCHAR (.P-1) NEQ ' '
DO P = .P - 1;
COMMAND_DESC[0] = .P - .COMMAND_DESC[1];
CH$FILL (0, ZERO_LENGTH, ZERO_AREA);
TPARSE_BLOCK[TPASL_STRINGCNT] = .COMMAND_DESC[0];
TPARSE_BLOCK[TPASL_STRINGPTR] = .COMMAND_DESC[1];
STATUS = LIB$TPARSE (TPARSE_BLOCK, STATE_TABLE, KEY_TABLE);
IF NOT .STATUS
THEN
BEGIN
IF .STATUS EQL LIB$SYNTAXERR
THEN STATUS = DSKQ$SYNTAX;
ERR_MESSAGE (.STATUS,
.TPARSE_BLOCK[TPASL_TOKENPTR] - .COMMAND_DESC[1],
.COMMAND_DESC[1],
```

```

: 638      P 0633 4      .TPARSE_BLOCK[TPASL_TOKENCNT],
: 639      P 0634 4      .TPARSE_BLOCK[TPASL_TOKENPTR],
: 640      P 0635 4      .TPARSE_BLOCK[TPASL_STRINGCNT] - .TPARSE_BLOCK[TPASL_TOKENCNT],
: 641      P 0636 4      .TPARSE_BLOCK[TPASL_STRINGPTR] + .TPARSE_BLOCK[TPASL_TOKENCNT]
: 642      P 0637 4      );
: 643      P 0638 4      END;
: 644      P 0639 4
: 645      P 0640 4      IF .CLEANUP_FLAGS[CLF_EXIT]      ! if EXIT command encountered
: 646      P 0641 4      THEN RETURN-1                  ! then exit DISK_QUOTA
: 647      P 0642 4
: 648      P 0643 4      END;                            ! end of command loop
: 649      P 0644 4
: 650      P 0645 4      1
: 651      P 0646 4      END;                            ! end of routine DISK_QUOTA
```

```

                                .TITLE DISKQUOTA
                                .IDENT  \V04-000\
                                .PSECT  _LIB$KEY1$,NOWRT, SHR, PIC,1

                                00000 :TPASKEYSTO
                                U.68:  .BLKB  0
                                44 44 41 00000 :TPASKEYST
                                U.70:  .ASCII \ADD\
                                FF 00003 :TPASKEYSTO
                                00004 :TPASKEYSTO
                                U.74:  .BLKB  0
                                45 54 41 45 52 43 00004 :TPASKEYST
                                U.76:  .ASCII \CREATE\
                                FF 0000A :TPASKEYSTO
                                0000B :TPASKEYSTO
                                U.81:  .BLKB  0
                                45 4C 42 41 53 49 44 0000B :TPASKEYST
                                U.83:  .ASCII \DISABLE\
                                FF 00012 :TPASKEYSTO
                                00013 :TPASKEYSTO
                                U.87:  .BLKB  0
                                45 4C 42 41 4E 45 00013 :TPASKEYST
                                U.89:  .ASCII \ENABLE\
                                FF 00019 :TPASKEYSTO
                                0001A :TPASKEYSTO
                                U.93:  .BLKB  0
                                54 49 58 45 0001A :TPASKEYST
                                U.95:  .ASCII \EXIT\
                                FF 0001E :TPASKEYSTO
                                0001F :TPASKEYSTO
                                U.100: .BLKB  0
                                50 4C 45 48 0001F :TPASKEYST
                                U.102: .ASCII \HELP\
                                FF 00023 :TPASKEYSTO
                                00024 :TPASKEYSTO
                                U.106: .BLKB  0
                                59 46 49 44 4F 4D 00024 :TPASKEYST
                                U.108: .ASCII \MODIFY\
                                FF 0002A :TPASKEYSTO
                                0002B :TPASKEYSTO
```

```
44 4C 49 55 42 45 52 0002B U.112: .BLKB 0
                                :TPASKEYST
                                U.114: .ASCII \REBUILD\
                                FF 00032 :.BYTE -1
                                00033 :TPASKEYSTO
45 56 4F 4D 45 52 00033 U.118: .BLKB 0
                                :TPASKEYST
                                U.120: .ASCII \REMOVE\
                                FF 00039 :.BYTE -1
                                0003A :TPASKEYSTO
57 4F 48 53 0003A U.124: .BLKB 0
                                :TPASKEYST
                                U.126: .ASCII \SHOW\
                                FF 0003E :.BYTE -1
                                0003F :TPASKEYSTO
45 53 55 0003F U.130: .BLKB 0
                                :TPASKEYST
                                U.132: .ASCII \USE\
                                FF 00042 :.BYTE -1
                                FF 00043 :TPASKEYFILL
                                U.140: .BYTE -1
                                00044 :TPASKEYSTO
41 54 4F 55 51 4D 52 45 50 00044 U.187: .BLKB 0
                                :TPASKEYST
                                U.189: .ASCII \PERMQUOTA\
                                FF 0004D :.BYTE -1
                                0004E :TPASKEYSTO
54 46 41 52 44 52 45 56 4F 0004E U.195: .BLKB 0
                                :TPASKEYST
                                U.197: .ASCII \OVERDRAFT\
                                FF 00057 :.BYTE -1
                                FF 00058 :TPASKEYFILL
                                U.205: .BYTE -1
                                00059 :TPASKEYSTO
41 54 4F 55 51 4D 52 45 50 00059 U.242: .BLKB 0
                                :TPASKEYST
                                U.244: .ASCII \PERMQUOTA\
                                FF 00062 :.BYTE -1
                                00063 :TPASKEYSTO
54 46 41 52 44 52 45 56 4F 00063 U.247: .BLKB 0
                                :TPASKEYST
                                U.249: .ASCII \OVERDRAFT\
                                FF 0006C :.BYTE -1
                                FF 0006D :TPASKEYFILL
                                U.254: .BYTE -1

                                .PSECT _LIB$STATES, NOWRT, SHR, PIC, 1

                                00000 STATE_TABLE::
                                :.BLKB 0
00000 00000 START: .BLKB 0
1100 00000 :TPASTYPE
                                U.71: .WORD 4352
0000* 00002 :TPASTARGET
                                U.73: .WORD <<U.72-U.73>-2>
9101 00004 :TPASTYPE
                                U.77: .WORD -28415
```


| | | | |
|-----------|-------|--------------|--------------------------|
| 00000000V | 00006 | :TPASACTION | |
| | | U.78: .LONG | <<ACT_CREATE-U.78>-4> |
| 0000* | 0000A | :TPASTARGET | |
| | | U.80: .WORD | <<U.79-U.80>-2> |
| 9102 | 0000C | :TPASTYPE | |
| | | U.84: .WORD | -28414 |
| 00000000V | 0000E | :TPASACTION | |
| | | U.85: .LONG | <<ACT_DISABLE-U.85>-4> |
| 0000* | 00012 | :TPASTARGET | |
| | | U.86: .WORD | <<U.79-U.86>-2> |
| 9103 | 00014 | :TPASTYPE | |
| | | U.90: .WORD | -28413 |
| 00000000V | 00016 | :TPASACTION | |
| | | U.91: .LONG | <<ACT_ENABLE-U.91>-4> |
| 0000* | 0001A | :TPASTARGET | |
| | | U.92: .WORD | <<U.79-U.92>-2> |
| 7104 | 0001C | :TPASTYPE | |
| | | U.96: .WORD | 28932 |
| 00000000* | 0001E | :TPASADDR | |
| | | U.97: .LONG | <<CLEANUP_FLAGS-U.97>-4> |
| 00000002 | 00022 | :TPASMASK | |
| | | U.98: .LONG | 2 |
| FFFF | 00026 | :TPASTARGET | |
| | | U.99: .WORD | -1 |
| 1105 | 00028 | :TPASTYPE | |
| | | U.103: .WORD | 4357 |
| 0000* | 0002A | :TPASTARGET | |
| | | U.105: .WORD | <<U.104-U.105>-2> |
| 1106 | 0002C | :TPASTYPE | |
| | | U.109: .WORD | 4358 |
| 0000* | 0002E | :TPASTARGET | |
| | | U.111: .WORD | <<U.110-U.111>-2> |
| 9107 | 00030 | :TPASTYPE | |
| | | U.115: .WORD | -28409 |
| 00000000V | 00032 | :TPASACTION | |
| | | U.116: .LONG | <<ACT_REBUILD-U.116>-4> |
| 0000* | 00036 | :TPASTARGET | |
| | | U.117: .WORD | <<U.79-U.117>-2> |
| 1108 | 00038 | :TPASTYPE | |
| | | U.121: .WORD | 4360 |
| 0000* | 0003A | :TPASTARGET | |
| | | U.123: .WORD | <<U.122-U.123>-2> |
| 1109 | 0003C | :TPASTYPE | |
| | | U.127: .WORD | 4361 |
| 0000* | 0003E | :TPASTARGET | |
| | | U.129: .WORD | <<U.128-U.129>-2> |
| 110A | 00040 | :TPASTYPE | |
| | | U.133: .WORD | 4362 |
| 0000* | 00042 | :TPASTARGET | |
| | | U.135: .WORD | <<U.134-U.135>-2> |
| 81F1 | 00044 | :TPASTYPE | |
| | | U.136: .WORD | -32271 |
| 00000000V | 00046 | :TPASACTION | |
| | | U.137: .LONG | <<INV_COMMAND-U.137>-4> |
| 15F7 | 0004A | :TPASTYPE | |
| | | U.138: .WORD | 5623 |
| FFFF | 0004C | :TPASTARGET | |

| | | | | |
|-----------|-------|--------------|---------------------|---|
| | | U.139: .WORD | -1 | : |
| | 0004E | :DO USE | | : |
| | | U.134: .BLKB | 0 | : |
| 9DF8 | 0004E | :TPASTYPE | | : |
| | | U.141: .WORD | -25096 | : |
| 0000* | 00050 | :TPASSUBEXP | | : |
| | | U.143: .WORD | <<U.142-U.143>-2> | : |
| 00000000V | 00052 | :TPASACTION | | : |
| | | U.144: .LONG | <<ACT_USE-U.144>-4> | : |
| 0000* | 00056 | :TPASTARGET | | : |
| | | U.145: .WORD | <<U.79-U.145>-2> | : |
| | 00058 | :DO ADD | | : |
| | | U.72: .BLKB | 0 | : |
| 19F8 | 00058 | :TPASTYPE | | : |
| | | U.146: .WORD | 6648 | : |
| 0000* | 0005A | :TPASSUBEXP | | : |
| | | U.148: .WORD | <<U.147-U.148>-2> | : |
| 0000* | 0005C | :TPASTARGET | | : |
| | | U.149: .WORD | <<U.72-U.149>-2> | : |
| 1DF8 | 0005E | :TPASTYPE | | : |
| | | U.150: .WORD | 7672 | : |
| 0G00* | 00060 | :TPASSUBEXP | | : |
| | | U.152: .WORD | <<U.151-U.152>-2> | : |
| 0000* | 00062 | :TPASTARGET | | : |
| | | U.154: .WORD | <<U.153-U.154>-2> | : |
| | 00064 | :DO ADD1 | | : |
| | | U.153: .BLKB | 0 | : |
| 19F8 | 00064 | :TPASTYPE | | : |
| | | U.155: .WORD | 6648 | : |
| 0000* | 00066 | :TPASSUBEXP | | : |
| | | U.156: .WORD | <<U.147-U.156>-2> | : |
| 0000* | 00068 | :TPASTARGET | | : |
| | | U.157: .WORD | <<U.153-U.157>-2> | : |
| 95F6 | 0006A | :TPASTYPE | | : |
| | | U.158: .WORD | -27146 | : |
| 00000000V | 0006C | :TPASACTION | | : |
| | | U.159: .LONG | <<ACT_ADD-U.159>-4> | : |
| 0000* | 00070 | :TPASTARGET | | : |
| | | U.160: .WORD | <<U.79-U.160>-2> | : |
| | 00072 | :DO MODIFY | | : |
| | | U.170: .BLKB | 0 | : |
| 19F8 | 00072 | :TPASTYPE | | : |
| | | U.161: .WORD | 6648 | : |
| 0000* | 00074 | :TPASSUBEXP | | : |
| | | U.162: .WORD | <<U.147-U.162>-2> | : |
| 0000* | 00076 | :TPASTARGET | | : |
| | | U.163: .WORD | <<U.110-U.163>-2> | : |
| 1DF8 | 00078 | :TPASTYPE | | : |
| | | U.164: .WORD | 7672 | : |
| 0000* | 0007A | :TPASSUBEXP | | : |
| | | U.165: .WORD | <<U.151-U.165>-2> | : |
| 0000* | 0007C | :TPASTARGET | | : |
| | | U.167: .WORD | <<U.166-U.167>-2> | : |
| | 0007E | :DO MODIFY1 | | : |
| | | U.166: .BLKB | 0 | : |
| 19F8 | 0007E | :TPASTYPE | | : |
| | | U.168: .WORD | 6648 | : |

```
0000* 00080 :TPASSUBEXP
              U.169: .WORD    <<U.147-U.169>-2>
0000* 00082 :TPASTARGET
              U.170: .WORD    <<U.166-U.170>-2>
95F6 00084 :TPASTYPE
              U.171: .WORD    -27146
00000000V 00086 :TPASACTION
              U.172: .LONG    <<ACT_MODIFY-U.172>-4>
0000* 0008A :TPASTARGET
              U.173: .WORD    <<U.79-U.173>-2>
              0008C :DO SHOW
              U.128: .BLKB    0
9DF8 0008C :TPASTYPE
              U.174: .WORD    -25096
0000* 0008E :TPASSUBEXP
              U.175: .WORD    <<U.151-U.175>-2>
00000000V 00090 :TPASACTION
              U.176: .LONG    <<ACT_SHOW-U.176>-4>
0000* 00094 :TPASTARGET
              U.177: .WORD    <<U.79-U.177>-2>
              00096 :DO REMOVE
              U.122: .BLKB    0
9DF8 00096 :TPASTYPE
              U.178: .WORD    -25096
0000* 00098 :TPASSUBEXP
              U.179: .WORD    <<U.151-U.179>-2>
00000000V 0009A :TPASACTION
              U.180: .LONG    <<ACT_REMOVE-U.180>-4>
0000* 0009E :TPASTARGET
              U.181: .WORD    <<U.79-U.181>-2>
              000A0 :MORE
              U.79: .BLKB    0
103B 000A0 :TPASTYPE
              U.182: .WORD    4155
0000* 000A2 :TPASTARGET
              U.183: .WORD    <<START-U.183>-2>
15F7 000A4 :TPASTYPE
              U.184: .WORD    5623
FFFF 000A6 :TPASTARGET
              U.185: .WORD    -1
              000AB :CMD SWIT
              U.147: .BLKB    0
042F 000AB :TPASTYPE
              U.186: .WORD    1071
710B 000AA :TPASTYPE
              U.190: .WORD    28939
00000000* 000AC :TPASADDR
              U.191: .LONG    <<UIC_FLAGS-U.191>-4>
00000008 000B0 :TPASMASK
              U.192: .LONG    8
0000* 000B4 :TPASTARGET
              U.194: .WORD    <<U.193-U.194>-2>
710C 000B6 :TPASTYPE
              U.198: .WORD    28940
00000000* 000B8 :TPASADDR
              U.199: .LONG    <<UIC_FLAGS-U.199>-4>
00000010 000BC :TPASMASK
```


| | | | | | |
|-----------|-------|---------------|------|------------------------|---|
| | | U.200: | LONG | 16 | : |
| 0000* | 000C0 | :TPASTARGET | | | : |
| | | U.202: | WORD | <<U.201-U.202>-2> | : |
| 85F1 | 000C2 | :TPASTYPE | | | : |
| | | U.203: | WORD | -31247 | : |
| 00000000V | 000C4 | :TPASACTION | | | : |
| | | U.204: | LONG | <<INV_SWITCH-U.204>-4> | : |
| | 000C8 | :DO PERMQUOTA | | | : |
| | | U.193: | BLKB | 0 | : |
| 043D | 000C8 | :TPASTYPE | | | : |
| | | U.206: | WORD | 1085 | : |
| 55F3 | 000CA | :TPASTYPE | | | : |
| | | U.207: | WORD | 22003 | : |
| 00000000* | 000CC | :TPASADDR | | | : |
| | | U.208: | LONG | <<PERM_VALUE-U.208>-4> | : |
| FFFF | 000D0 | :TPASTARGET | | | : |
| | | U.209: | WORD | -1 | : |
| | 000D2 | :DO OVERDRAFT | | | : |
| | | U.201: | BLKB | 0 | : |
| 043D | 000D2 | :TPASTYPE | | | : |
| | | U.210: | WORD | 1085 | : |
| 55F3 | 000D4 | :TPASTYPE | | | : |
| | | U.211: | WORD | 22003 | : |
| 00000000* | 000D6 | :TPASADDR | | | : |
| | | U.212: | LONG | <<OVER_VALUE-U.212>-4> | : |
| FFFF | 000DA | :TPASTARGET | | | : |
| | | U.213: | WORD | -1 | : |
| | 000DC | :DEV SPEC | | | : |
| | | U.142: | BLKB | 0 | : |
| 05F1 | 000DC | :TPASTYPE | | | : |
| | | U.214: | WORD | 1521 | : |
| 103A | 000DE | :TPASTYPE | | | : |
| | | U.215: | WORD | 4154 | : |
| FFFF | 000E0 | :TPASTARGET | | | : |
| | | U.216: | WORD | -1 | : |
| 15F6 | 000E2 | :TPASTYPE | | | : |
| | | U.217: | WORD | 5622 | : |
| FFFF | 000E4 | :TPASTARGET | | | : |
| | | U.218: | WORD | -1 | : |
| | 000E6 | :UIC | | | : |
| | | U.151: | BLKB | 0 | : |
| 55EC | 000E6 | :TPASTYPE | | | : |
| | | U.219: | WORD | 21996 | : |
| 00000000* | 000E8 | :TPASADDR | | | : |
| | | U.220: | LONG | <<UIC_VALUE-U.220>-4> | : |
| FFFF | 000EC | :TPASTARGET | | | : |
| | | U.221: | WORD | -1 | : |
| | 000EE | :DO HELP | | | : |
| | | U.104: | BLKB | 0 | : |
| 91F0 | 000EE | :TPASTYPE | | | : |
| | | U.222: | WORD | -28176 | : |
| 00000000V | 000F0 | :TPASACTION | | | : |
| | | U.223: | LONG | <<SAVE_KEY-U.223>-4> | : |
| 0000* | 000F4 | :TPASTARGET | | | : |
| | | U.224: | WORD | <<U.104-U.224>-2> | : |
| 99F8 | 000F6 | :TPASTYPE | | | : |
| | | U.225: | WORD | -26120 | : |

```
0000* 000F8 :TPASSUBEXP
00000000V 000FA :TPASACTION
0000* 000FE :TPASTARGET
902A 00100 :TPASTYPE
00000000V 00102 :TPASACTION
0000* 00106 :TPASTARGET
99F8 00108 :TPASTYPE
0000* 0010A :TPASSUBEXP
00000000V 0010C :TPASACTION
0000* 00110 :TPASTARGET
95F6 00112 :TPASTYPE
00000000V 00114 :TPASACTION
0000* 00118 :TPASTARGET
0011A :DO QUALIFIER
042F 0011A :TPASTYPE
110D 0011C :TPASTYPE
FFFF 0011E :TPASTARGET
110E 00120 :TPASTYPE
FFFF 00122 :TPASTARGET
15F0 00124 :TPASTYPE
FFFF 00126 :TPASTARGET
00128 :ELIPSIS
042E 00128 :TPASTYPE
042E 0012A :TPASTYPE
142E 0012C :TPASTYPE
FFFF 0012E :TPASTARGET
```

.PSECT _LIB\$KEYOS,NOWRT, SHR, PIC,1

00000 KEY_TABLE::
.BLKB 0

```
00000 :TPASKEY0
      U.67: .BLKB 0
0000* 00000 :TPASKEY
      U.69: .WORD <U.68-U.67>
0000* 00002 :TPASKEY
      U.75: .WORD <U.74-U.67>
0000* 00004 :TPASKEY
      U.82: .WORD <U.81-U.67>
0000* 00006 :TPASKEY
      U.88: .WORD <U.87-U.67>
0000* 00008 :TPASKEY
      U.94: .WORD <U.93-U.67>
0000* 0000A :TPASKEY
      U.101: .WORD <U.100-U.67>
0000* 0000C :TPASKEY
      U.107: .WORD <U.106-U.67>
0000* 0000E :TPASKEY
      U.113: .WORD <U.112-U.67>
0000* 00010 :TPASKEY
      U.119: .WORD <U.118-U.67>
0000* 00012 :TPASKEY
      U.125: .WORD <U.124-U.67>
0000* 00014 :TPASKEY
      U.131: .WORD <U.130-U.67>
0000* 00016 :TPASKEY
      U.188: .WORD <U.187-U.67>
0000* 00018 :TPASKEY
      U.196: .WORD <U.195-U.67>
0000* 0001A :TPASKEY
      U.243: .WORD <U.242-U.67>
0000* 0001C :TPASKEY
      U.248: .WORD <U.247-U.67>
```

.PSECT \$MSG_INDEX,NOWRT,NOEXE,2

```
00000 MESSAGE_TABLE:
      .BLKB 0
00000000* 00000 :MSG_INDEX
      U.2: .ADDRESS U.1
00000000* 00004 :MSG_INDEX
      U.4: .ADDRESS U.3
00000000* 00008 :MSG_INDEX
      U.6: .ADDRESS U.5
00000000* 0000C :MSG_INDEX
      U.8: .ADDRESS U.7
00000000* 00010 :MSG_INDEX
      U.10: .ADDRESS U.9
00000000* 00014 :MSG_INDEX
      U.12: .ADDRESS U.11
00000000* 00018 :MSG_INDEX
      U.14: .ADDRESS U.13
00000000* 0001C :MSG_INDEX
      U.16: .ADDRESS U.15
00000000* 00020 :MSG_INDEX
      U.18: .ADDRESS U.17
00000000* 00024 :MSG_INDEX
      U.20: .ADDRESS U.19
```



```
00000000' 00028 :MSG_INDEX
                U.22: ADDRESS U.21
00000000' 0002C :MSG_INDEX
                U.24: ADDRESS U.23
00000000' 00030 :MSG_INDEX
                U.26: ADDRESS U.25
00000000' 00034 :MSG_INDEX
                U.28: ADDRESS U.27
00000000' 00038 :MSG_INDEX
                U.30: ADDRESS U.29
00000000' 0003C :MSG_INDEX
                U.32: ADDRESS U.31
00000000' 00040 :MSG_INDEX
                U.34: ADDRESS U.33
00000000' 00044 :MSG_INDEX
                U.36: ADDRESS U.35
00000000' 00048 :MSG_INDEX
                U.38: ADDRESS U.37
00000000' 0004C :MSG_INDEX
                U.40: ADDRESS U.39
00000000' 00050 :MSG_INDEX
                U.42: ADDRESS U.41
00000000' 00054 :MSG_INDEX
                U.44: ADDRESS U.43
00000000' 00058 :MSG_INDEX
                U.46: ADDRESS U.45
00000000' 0005C :MSG_INDEX
                U.48: ADDRESS U.47
00000000' 00060 :MSG_INDEX
                U.50: ADDRESS U.49
00000000' 00064 :MSG_INDEX
                U.52: ADDRESS U.51
00000000' 00068 :MSG_INDEX
                U.54: ADDRESS U.53
00000000' 0006C :MSG_INDEX
                U.56: ADDRESS U.55
00000000' 00070 :MSG_INDEX
                U.58: ADDRESS U.57
00000000' 00074 :MSG_INDEX
                U.60: ADDRESS U.59
00000000' 00078 :MSG_INDEX
                U.62: ADDRESS U.61
00000000' 0007C :MSG_INDEX
                U.64: ADDRESS U.63
00000000' 00080 :MSG_INDEX
                U.66: ADDRESS U.65
```

.PSECT \$MSG_TEXT,NOWRT,NOEXE,0

00000 MESSAGE_TEXT:
BLKB 0

```
2C 00 00000 :MSG_TEXT
                U.1: .BYTE 0,44
2D 51 4B 53 49 44 25 00002 .ASCII \DISKQ-\
                46 00009 .ASCII \F\
                2D 0000A .ASCII \-
52 52 45 5F 44 4D 43 0000B .ASCII \CMD_ERR\
```

[illegible]

```

                                32 06 00138 :MSG TEXT
                                U.15: .BYTE 0, 50
                                2D 51 4B 53 49 44 25 0013A .ASCII \DISKQ-\
                                45 00141 .ASCII \E\
                                2D 00142 .ASCII \- \
                                5B 41 54 4E 59 53 00143 .ASCII \SYNTAX\
                                20 2C 00149 .ASCII \, \
                                20 64 6E 61 6D 6D 6F 63 0014B .ASCII \command syntax error!/:AD\<92>\:AD\<92>
5C 44 41 21 5C 44 41 21 2F 21 72 6F 72 72 65 0015A .ASCII
                                44 41 21 00169 .ASCII \:AD\
                                2F 00 0016C :MSG TEXT
                                U.15: .BYTE 0, 47
                                2D 51 4B 53 49 44 25 0016E .ASCII \DISKQ-\
                                45 00175 .ASCII \E\
                                2D 00176 .ASCII \- \
                                4C 41 43 4F 4C 4E 4F 4E 00177 .ASCII \NONLOCAL\
                                20 2C 0017F .ASCII \, \
                                61 20 74 6F 6E 20 73 69 20 65 63 69 76 65 64 00181 .ASCII \device is not a local device\
                                65 63 69 76 65 64 20 6C 61 63 6F 6C 20 00190
                                0019D .BLKB 3
                                34 00 001A0 :MSG TEXT
                                U.17: .BYTE 0, 52
                                2D 51 4B 53 49 44 25 001A2 .ASCII \DISKQ-\
                                45 001A9 .ASCII \E\
                                2D 001AA .ASCII \- \
                                4E 41 52 54 4F 4E 001AB .ASCII \NOTRAN\
                                20 2C 001B1 .ASCII \, \
                                73 69 20 65 6D 61 6E 20 6C 61 63 69 67 6F 6C 001B3 .ASCII \logical name is recursively defined\
                                65 64 20 79 6C 65 76 69 73 72 75 63 65 72 20 001C2
                                64 65 6E 69 66 001D1
                                001D6 .BLKB 2
                                2F 00 001D8 :MSG TEXT
                                U.19: .BYTE 0, 47
                                2D 51 4B 53 49 44 25 001DA .ASCII \DISKQ-\
                                45 001E1 .ASCII \E\
                                2D 001E2 .ASCII \- \
                                45 43 49 56 45 44 4F 4E 001E3 .ASCII \NODEVICE\
                                20 2C 001EB .ASCII \, \
                                65 72 72 75 63 20 65 63 69 76 65 64 20 6F 6E 001ED .ASCII \no device currently selected\
                                64 65 74 63 65 6C 65 73 20 79 6C 74 6E 001FC
                                00209 .BLKB 3
                                2C 00 0020C :MSG TEXT
                                U.21: .BYTE 0, 44
                                2D 51 4B 53 49 44 25 0020E .ASCII \DISKQ-\
                                45 00215 .ASCII \E\
                                2D 00216 .ASCII \- \
                                52 52 45 54 41 45 52 43 00217 .ASCII \CREATERR\
                                20 2C 0021F .ASCII \, \
                                20 67 6E 69 74 61 65 72 63 20 72 6F 72 72 65 00221 .ASCII \error creating quota file\
                                65 6C 69 66 20 61 74 6F 75 71 00230
                                0023A .BLKB 2
                                2F 00 0023C :MSG TEXT
                                U.23: .BYTE 0, 47
                                2D 51 4B 53 49 44 25 0023E .ASCII \DISKQ-\
                                45 00245 .ASCII \E\
                                2D 00246 .ASCII \- \
                                52 52 45 54 49 4E 49 00247 .ASCII \INITERR\
```



```
7A 69 6C 61 69 74 69 6E 69 20 72 6F 72 20 2C 0024E .ASCII \. \
65 6C 69 66 20 61 74 6F 75 71 20 67 6E 69 00250 .ASCII \error initializing quota file\
0025F
0026D .BLKB 3
2A 00 00270 :MSG TEXT
U.25: .BYTE 0, 42
00272 .ASCII \DISKQ-\
00279 .ASCII \E\
0027A .ASCII \-\
0027B .ASCII \CLOERR\
00282 .ASCII \. \
00284 .ASCII \error closing quota file\
00293
0029C :MSG TEXT
U.27: .BYTE 0, 44
0029E .ASCII \DISKQ-\
002A5 .ASCII \E\
002A6 .ASCII \-\
002A7 .ASCII \ACTERR\
002AD .ASCII \. \
002AF .ASCII \failed to enable quota file\
002BE
002CA .BLKB 2
2E 00 002CC :MSG TEXT
U.29: .BYTE 0, 46
002CE .ASCII \DISKQ-\
002D5 .ASCII \E\
002D6 .ASCII \-\
002D7 .ASCII \DACTERR\
002DE .ASCII \. \
002E0 .ASCII \failed to disable quota file\
002EF
002FC :MSG TEXT
U.31: .BYTE 0, 47
002FE .ASCII \DISKQ-\
00305 .ASCII \E\
00306 .ASCII \-\
00307 .ASCII \ADDERR\
0030D .ASCII \. \
0030F .ASCII \failed to add quota file entry\
0031E
0032D .BLKB 3
34 00 00330 :MSG TEXT
U.33: .BYTE 0, 52
00332 .ASCII \DISKQ-\
00339 .ASCII \E\
0033A .ASCII \-\
0033B .ASCII \REMOVERR\
00343 .ASCII \. \
00345 .ASCII \failed to remove quota file entry\
00354
00363 .BLKB 2
35 00 00368 :MSG TEXT
U.35: .BYTE 0, 53
0036A .ASCII \DISKQ-\
00371 .ASCII \E\
```

```

      52 52 45 59 46 49 44 4F 4D 00372 .ASCII \- \
      20 2C 00373 .ASCII \MODIFYERR\
66 69 64 6F 6D 20 6F 74 20 64 65 6C 69 61 66 0037C .ASCII \, \
6E 65 20 65 6C 69 66 20 61 74 6F 75 71 20 79 0037E .ASCII \failed to modify quota file entry\
      79 72 74 0038D
      0039C
      0039F
      33 00 003A0 :MSG TEXT BLKB 1
      U.37: .BYTE 0 51
      2D 51 4B 53 49 44 25 003A2 .ASCII \DISKQ-\
      45 003A9 .ASCII \E\
      2D 003AA .ASCII \- \
      52 52 45 4E 49 4D 41 58 45 003AB .ASCII \EXAMINERR\
      20 2C 003B4 .ASCII \, \
20 65 6E 69 6D 61 78 65 20 74 6F 6E 6E 61 63 003B6 .ASCII \cannot examine quota file entry\
72 74 6E 65 20 65 6C 69 66 20 61 74 6F 75 71 003C5
      79 003D4
      003D5
      2F 03 003D8 :MSG TEXT BLKB 3
      U.39: .BYTE 3 47
      2D 51 4B 53 49 44 25 003DA .ASCII \DISKQ-\
      49 003E1 .ASCII \I\
      2D 003E2 .ASCII \- \
      45 53 55 4E 49 003E3 .ASCII \INUSE\
      20 2C 003E8 .ASCII \, \
21 20 73 61 68 20 5D 57 4F 21 2C 57 4F 21 5B 003EA .ASCII \[!OW,!OW] has !UL blocks in use\
73 75 20 6E 69 20 73 6B 63 6F 6C 62 20 4C 55 003F9
      65 00408
      00409
      27 00 0040C :MSG TEXT BLKB 3
      U.41: .BYTE 0 39
      2D 51 4B 53 49 44 25 0040E .ASCII \DISKQ-\
      45 00415 .ASCII \E\
      2D 00416 .ASCII \- \
      52 52 45 4B 43 4F 4C 00417 .ASCII \LOCKERR\
      20 2C 0041E .ASCII \, \
20 6B 63 6F 6C 20 6F 74 20 64 65 6C 69 61 66 00420 .ASCII \failed to lock volume\
      65 6D 75 6C 6F 76 0042F
      00435
      2B 00 00438 :MSG TEXT BLKB 3
      U.43: .BYTE 0 43
      2D 51 4B 53 49 44 25 0043A .ASCII \DISKQ-\
      45 00441 .ASCII \E\
      2D 00442 .ASCII \- \
      52 52 45 4B 43 4F 4C 4E 55 00443 .ASCII \UNLOCKERR\
      20 2C 0044C .ASCII \, \
63 6F 6C 6E 75 20 6F 74 20 64 65 6C 69 61 66 0044E .ASCII \failed to unlock volume\
      65 6D 75 6C 6F 76 20 6B 0045D
      00465
      3B 00 00468 :MSG TEXT BLKB 3
      U.45: .BYTE 0 59
      2D 51 4B 53 49 44 25 0046A .ASCII \DISKQ-\
      45 00471 .ASCII \E\
      2D 00472 .ASCII \- \
      53 4C 4F 56 58 41 4D 00473 .ASCII \MAXVOLS\
      20 2C 0047A .ASCII \, \
20 73 61 68 20 74 65 73 20 65 6D 75 6C 6F 76 0047C .ASCII \volume set has too many volumes to handl\

```

| | | | | | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|-----------|--|--|--|
| 65 | 6D | 75 | 6C | 6F | 76 | 20 | 79 | 6E | 61 | 6D | 20 | 6F | 6F | 74 | 004BB | | | | |
| | | | | | 6C | 64 | 6E | 61 | 68 | 20 | 6F | 74 | 20 | 73 | 0049A | | | | |
| | | | | | | | | | | | | | | 65 | 004A4 | .ASCII | \e\ | | |
| | | | | | | | | | | | | | | | 004A5 | .BLKB | 3 | | |
| | | | | | | | | | | | | | 46 | 01 | 004A8 | :MSG_TEXT | | | |
| | | | | | | | | | | | | | | | U.47: | .BYTE | 1, 70 | | |
| | | | | | | | 2D | 51 | 4B | 53 | 49 | 44 | 25 | | 004AA | .ASCII | \%DISKQ-\ | | |
| | | | | | | | | | | | | | 45 | | 004B1 | .ASCII | \E\ | | |
| | | | | | | | | | | | | | 2D | | 004B2 | .ASCII | \-\ | | |
| | | | | | 46 | 58 | 45 | 44 | 4E | 49 | 43 | 43 | 41 | | 004B3 | .ASCII | \ACCINDEX\ | | |
| | | | | | | | | | | | | | 20 | 2C | 004BC | .ASCII | \ \ | | |
| 73 | 65 | 63 | 63 | 61 | 20 | 6F | 74 | 20 | 64 | 65 | 6C | 69 | 61 | 66 | 004BE | .ASCII | \failed to access index file on relative \ | | |
| 6E | 6F | 20 | 65 | 6C | 69 | 66 | 20 | 78 | 65 | 64 | 6E | 69 | 20 | 73 | 004CD | | | | |
| | | | | | 20 | 65 | 76 | 69 | 74 | 61 | 6C | 65 | 72 | 20 | 004DC | | | | |
| | | | | | 57 | 55 | 21 | 20 | 65 | 6D | 75 | 6C | 6F | 76 | 004E6 | .ASCII | \volume !UW\ | | |
| | | | | | | | | | | | | | 2E | 00 | 004F0 | :MSG_TEXT | | | |
| | | | | | | | | | | | | | | | U.49: | .BYTE | 0, 46 | | |
| | | | | | | | 2D | 51 | 4B | 53 | 49 | 44 | 25 | | 004F2 | .ASCII | \%DISKQ-\ | | |
| | | | | | | | | | | | | | 45 | | 004F9 | .ASCII | \E\ | | |
| | | | | | | | | | | | | | 2D | | 004FA | .ASCII | \-\ | | |
| | | | | | | | 45 | 4C | 49 | 46 | 51 | 43 | 43 | 41 | 004FB | .ASCII | \ACQFILE\ | | |
| | | | | | | | | | | | | | 20 | 2C | 00503 | .ASCII | \ \ | | |
| 73 | 65 | 63 | 63 | 61 | 20 | 6F | 74 | 20 | 64 | 65 | 6C | 69 | 61 | 66 | 00505 | .ASCII | \failed to access quota file\ | | |
| | | | 65 | 6C | 69 | 66 | 20 | 61 | 74 | 6F | 75 | 71 | 20 | 73 | 00514 | | | | |
| | | | | | | | | | | | | | 30 | 00 | 00520 | :MSG_TEXT | | | |
| | | | | | | | | | | | | | | | U.51: | .BYTE | 0, 48 | | |
| | | | | | | | 2D | 51 | 4B | 53 | 49 | 44 | 25 | | 00522 | .ASCII | \%DISKQ-\ | | |
| | | | | | | | | | | | | | 45 | | 00529 | .ASCII | \E\ | | |
| | | | | | | | | | | | | | 2D | | 0052A | .ASCII | \-\ | | |
| | | | | | 52 | 52 | 45 | 52 | 41 | 54 | 4F | 55 | 51 | | 0052B | .ASCII | \QUOTARERR\ | | |
| | | | | | | | | | | | | | 20 | 2C | 00534 | .ASCII | \ \ | | |
| 69 | 64 | 61 | 65 | 72 | 20 | 72 | 6F | 72 | 72 | 65 | 20 | 4F | 2F | 49 | 00536 | .ASCII | \I/O error reading quota file\ | | |
| | | 65 | 6C | 69 | 66 | 20 | 61 | 74 | 6F | 75 | 71 | 20 | 67 | 6E | 00545 | | | | |
| | | | | | | | | | | | | | | | 00552 | .BLKB | 2 | | |
| | | | | | | | | | | | | | 4E | 01 | 00554 | :MSG_TEXT | | | |
| | | | | | | | | | | | | | | | U.53: | .BYTE | 1, 78 | | |
| | | | | | | | 2D | 51 | 4B | 53 | 49 | 44 | 25 | | 00556 | .ASCII | \%DISKQ-\ | | |
| | | | | | | | | | | | | | 45 | | 0055D | .ASCII | \E\ | | |
| | | | | | | | | | | | | | 2D | | 0055E | .ASCII | \-\ | | |
| | | | | | 52 | 52 | 45 | 50 | 41 | 4D | 54 | 49 | 42 | | 0055F | .ASCII | \BITMAPERR\ | | |
| | | | | | | | | | | | | | 20 | 2C | 00568 | .ASCII | \ \ | | |
| 69 | 64 | 61 | 65 | 72 | 20 | 72 | 6F | 72 | 72 | 65 | 20 | 4F | 2F | 49 | 0056A | .ASCII | \I/O error reading index file bitmap on r\ | | |
| 62 | 20 | 65 | 6C | 69 | 66 | 20 | 78 | 65 | 64 | 6E | 69 | 20 | 67 | 6E | 00579 | | | | |
| | | | | | 72 | 20 | 6E | 6F | 20 | 70 | 61 | 6D | 74 | 69 | 00588 | | | | |
| 20 | 65 | 6D | 75 | 6C | 6F | 76 | 20 | 65 | 76 | 69 | 74 | 61 | 6C | 65 | 00592 | .ASCII | \relative volume !UW\ | | |
| | | | | | | | | | | | | 57 | 55 | 21 | 005A1 | | | | |
| | | | | | | | | | | | | 4C | 02 | | 005A4 | :MSG_TEXT | | | |
| | | | | | | | | | | | | | | | U.55: | .BYTE | 2, 76 | | |
| | | | | | | | 2D | 51 | 4B | 53 | 49 | 44 | 25 | | 005A6 | .ASCII | \%DISKQ-\ | | |
| | | | | | | | | | | | | | 57 | | 005AD | .ASCII | \W\ | | |
| | | | | | | | | | | | | | 2D | | 005AE | .ASCII | \-\ | | |
| | | | | | 52 | 52 | 45 | 52 | 45 | 44 | 41 | 45 | 48 | | 005AF | .ASCII | \HEADERERR\ | | |
| | | | | | | | | | | | | | 20 | 2C | 005B8 | .ASCII | \ \ | | |
| 69 | 64 | 61 | 65 | 72 | 20 | 72 | 6F | 72 | 72 | 65 | 20 | 4F | 2F | 49 | 005BA | .ASCII | \I/O error reading file header !UL on rel\ | | |
| 20 | 72 | 65 | 64 | 61 | 65 | 68 | 20 | 65 | 6C | 69 | 66 | 20 | 67 | 6E | 005C9 | | | | |
| | | | | | 6C | 65 | 72 | 20 | 6E | 6F | 20 | 4C | 55 | 21 | 005D8 | | | | |
| 55 | 21 | 20 | 65 | 6D | 75 | 6C | 6F | 76 | 20 | 65 | 76 | 69 | 74 | 61 | 005E2 | .ASCII | \ative volume !UW\ | | |

```

57 005F1
34 00 005F2
      005F4 :MSG-TEXT BLKB 2
      U.57: .BYTE 0, 52
      .ASCII \DISKQ-\
      .ASCII \E\
      .ASCII \- \
      .ASCII \MEMALLOC\
      .ASCII \, \
      .ASCII \cannot allocate sufficient memory\
65 74 61 63 6F 6C 6C 61 20 74 6F 6E 6E 61 63 00607
6D 65 6D 20 74 6E 65 69 63 69 66 66 75 73 20 00609
      79 72 6F 00618
      00627
      0062A
      44 01 0062C :MSG-TEXT BLKB 2
      U.59: .BYTE 1, 68
      .ASCII \DISKQ-\
      .ASCII \E\
      .ASCII \- \
      .ASCII \HOMEBLOCK\
      .ASCII \, \
      .ASCII \failed to read home block on relative vol\
20 64 61 65 72 20 6F 74 20 64 65 6C 69 61 66 00642
72 20 6E 6F 20 6B 63 6F 6C 62 20 65 6D 6F 68 00651
      6F 76 20 65 76 69 74 61 6C 65 00660
      57 55 21 20 65 6D 75 6C 0066A
      00672
      32 01 00674 :MSG-TEXT BLKB 2
      U.61: .BYTE 1, 50
      .ASCII \DISKQ-\
      .ASCII \E\
      .ASCII \- \
      .ASCII \HELP_INIT\
      .ASCII \, \
      .ASCII \failed help library index init\
62 69 6C 20 70 6C 65 68 20 64 65 6C 69 61 66 00688
74 69 6E 69 20 78 65 64 6E 69 20 79 72 61 72 0068A
      2F 01 00699
      006A8 :MSG-TEXT BLKB 3
      U.63: .BYTE 1, 47
      .ASCII \DISKQ-\
      .ASCII \E\
      .ASCII \- \
      .ASCII \HELP_OPEN\
      .ASCII \, \
      .ASCII \failed to open help library\
20 6E 65 70 6F 20 6F 74 20 64 65 6C 69 61 66 006BE
79 72 61 72 62 69 6C 20 70 6C 65 68 006CD
      2E 01 006D9
      006DC :MSG-TEXT BLKB 3
      U.65: .BYTE 1, 46
      .ASCII \DISKQ-\
      .ASCII \E\
      .ASCII \- \
      .ASCII \HELP_TEXT\
      .ASCII \, \
      .ASCII \failed to access help text\
73 65 63 63 61 20 6F 74 20 64 65 6C 69 61 66 006F0
74 78 65 74 20 70 6C 65 68 20 73 006F2
      00701
      .PSECT SPLITS, NOWRT, NOEXE, 2
OE OD OC OB OA O9 O8 O7 O6 O5 O4 O3 O2 O1 O0 00000 P.AAC: .BYTE 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, - ;
```


L 11
13-Sep-1984 23:38:38
14-Sep-1984 12:19:46VAX-11 Bliss-32 V4.0-742
[DISKQ.SRC]DISKQUOTA.B32;1Page 30
(6)

```
001C0 KEY_VECTOR: .BLKB 4
001F8 KEY_INDEX: .BLKB 56
001FC ZERO_END: .BLKB 4
00000020 001FC SRCREC_DESC: .BLKB 0
00000000' 00200 .LONG 32
00000020 00204 DSTREC_DESC: .ADDRESS SRC_REC
00000000' 00208 .LONG 32
00000040 0020C QFIB_DESC: .ADDRESS DST_REC
00000000' 00210 .LONG 64
00000002 00000008 00214 TPARSE_BLOCK: .ADDRESS QUOTA_FIB
0021C .LONG 8 2
00238 P.AAB: .BLKB 28
00243 .ASCII \QUOTA.SYS;1\
00000008 00244 P.AAA: .BLKB 1
00000000' 00248 .LONG 11
00000000' .ADDRESS P.AAB

UIC VALUE= SRC_REC+4
PERM VALUE= SRC_REC+12
OVER VALUE= SRC_REC+16
QFILE NAME= P.AAA
UPCASE_TABLE= P.AAC
.EXTRN LIB$SYNTAXERR, LIB$GET_INPUT
.EXTRN LIB$TPARSE, SYSSDCLEXH
.PSECT $CODE$,NOWRT,2
.ENTRY DISK QUOTA, Save R2,R3,R4,R5,R6,R7,R8,R9,- R10,R11
MOVAB LIB$SIGNAL, R11
MOVAB COMMAND_DESC, R10
MOVAL 98, (FP)
CALLS #0, USE_DEFAULT
MOVAB EXIT_HANDLER, EXIT_HNDLR_DESC+4
MOVL #1, EXIT_HNDLR_DESC+8
MOVAB EXIT_HNDLR_DESC+16, EXIT_HNDLR_DESC+12
PUSHAB EXIT_HNDLR_DESC
CALLS #1, SYSSDCLEXH
MOVZBL #132, COMMAND_DESC
PUSHAB P.AAD
PUSHL R10
CALLS #2, LIB$GET_INPUT
MOVL R0, STATUS
BLBS STATUS, 38
CMPL STATUS, #98938
BEQL 28
PUSHL STATUS
PUSHL #4521984
CALLS #2, LIB$SIGNAL
BRW 8$

0335
0374
0583
0589
0590
0591
0593
0604
0605
0606
0609
0610
0611
```

31 3B 53 59 53 2E 41 54 4F 55 51

00000000V 14 18 1C

00000000G 6A

00000000G 59 17

0001827A 8F

00450000 6B

0088 31 00064 28:

| | | | | | | | | | | |
|-----------|------|------|----------|------|----|-------|-------|--------|--|------|
| 00000000* | EF | 00 | 56 | 6A | 7D | 00067 | 38: | MOVQ | COMMAND_DESC, R6 | 0614 |
| | | | 67 | 56 | 2E | 0006A | | MOVTC | R6, (R7), #0, UPCASE_TABLE, R6, (R7) | 0615 |
| | | 58 | 67 | 56 | | 00073 | | | | |
| | | | 56 | 57 | C1 | 00075 | | ADDL3 | R7, R6, P | 0616 |
| | | | 20 | 57 | 91 | 00079 | 48: | CMPB | -1(P), #32 | 0617 |
| | | | | 04 | 12 | 0007D | | BNEQ | 58 | |
| | | | | 58 | 07 | 0007F | | DECL | P | 0618 |
| | | | | F6 | 11 | 00081 | | BRB | 48 | |
| 00C4 | BF | 6A | 58 | 57 | C3 | 00083 | 58: | SUBL3 | R7, P, COMMAND_DESC | 0619 |
| | | 00 | 6E | 00 | 2C | 00087 | | MOVCS | #0, (SP), #0, #196, ZERO_AREA | 0621 |
| | | | | AA | | 0008E | | | | |
| | | 0108 | CA | 6A | 7D | 00090 | | MOVQ | COMMAND_DESC, TPARSE_BLOCK+8 | 0622 |
| | | | | EF | 9F | 00095 | | PUSHAB | KEY_TABLE | 0624 |
| | | | | EF | 9F | 0009B | | PUSHAB | STATE_TABLE | |
| | | | | CA | 9F | 000A1 | | PUSHAB | TPARSE_BLOCK | |
| | | | | 03 | FB | 000A5 | | CALLS | #3, LIB\$TPARSE | |
| 00000000G | | 00 | | 50 | D0 | 000AC | | MOVL | R0, STATUS | |
| | | 59 | | 59 | E8 | 000AF | | BLBS | STATUS, 78 | 0625 |
| 00000000G | | 8F | | 59 | D1 | 000B2 | | CMPL | STATUS, #LIB\$_SYNTAXERR | 0628 |
| | | | | 07 | 12 | 000B9 | | BNEQ | 68 | |
| | | 59 | 00450030 | 8F | D0 | 000BB | | MOVL | #4522032, STATUS | 0629 |
| | | 50 | 0110 | CA | D0 | 000C2 | 68: | MOVL | TPARSE_BLOCK+16, R0 | 0637 |
| | | | 010C | DA40 | 9F | 000C7 | | PUSHAB | @TPARSE_BLOCK+12[R0] | |
| 7E | 0108 | CA | | 50 | C3 | 000CC | | SUBL3 | R0, TPARSE_BLOCK+8, -(SP) | |
| | | | | CA | DD | 000D2 | | PUSHL | TPARSE_BLOCK+20 | |
| | | | | 50 | DD | 000D6 | | PUSHL | R0 | |
| | | | | 04 | AA | 000D8 | | PUSHL | COMMAND_DESC+4 | |
| 7E | 0114 | CA | | 04 | AA | C3 | 000DB | SUBL3 | COMMAND_DESC+4, TPARSE_BLOCK+20, -(SP) | |
| | | | | 59 | DD | 000E2 | | PUSHL | STATUS | |
| | | 6B | | 07 | FB | 000E4 | | CALLS | #7, LIB\$SIGNAL | |
| 03 | 24 | AA | | 01 | E0 | 000E7 | 78: | BBS | #1, CLEANUP_FLAGS, 88 | 0640 |
| | | | | FF48 | 31 | 000EC | | BRW | 18 | |
| | | 50 | | 01 | D0 | 000EF | 88: | MOVL | #1, R0 | 0646 |
| | | | | 04 | | 000F2 | | RET | | |
| | | | | 0000 | | 000F3 | 98: | .WORD | Save nothing | 0374 |
| | | | | 7E | D4 | 000F5 | | CLRL | -(SP) | |
| | | | | 5E | DD | 000F7 | | PUSHL | SP | |
| | | 7E | | AC | 7D | 000F9 | | MOVQ | 4(AP), -(SP) | |
| 00000000V | EF | | 04 | 03 | FB | 000FD | | CALLS | #3, MAIN_HANDLER | |
| | | | | 04 | | 00104 | | RET | | |

; Routine Size: 261 bytes, Routine Base: \$CODE\$ + 0000

```

653      0647 1  |
654      0648 1  | Minor action routines to help out with parsing
655      0649 1  |
656      0650 1  |
657      0651 1  |
658      0652 1  | Give invalid command status
659      0653 1  |
660      0654 1  |
661      0655 1  | ROUTINE INV_COMMAND =
662      0656 1  |
663      0657 2  | BEGIN
664      0658 2  | TPARSE_ARGS;
665      0659 2  |
666      P 0660 2  | ERR_EXIT ((IF .TPARSE_BLOCK[TPASV_ambiguous]
667      P 0661 2  | THEN DSKQ$-AMB_CMD
668      P 0662 2  | ELSE DSKQ$-INV_CMD)
669      P 0663 2  | .TPARSE_BLOCK[TPASL_TOKENPTR] - .COMMAND_DESC[1],
670      P 0664 2  | .COMMAND_DESC[1],
671      P 0665 2  | .TPARSE_BLOCK[TPASL_TOKENCNT],
672      P 0666 2  | .TPARSE_BLOCK[TPASL_TOKENPTR],
673      P 0667 2  | .TPARSE_BLOCK[TPASL_STRINGCNT],
674      P 0668 2  | .TPARSE_BLOCK[TPASL_STRINGPTR]
675      0669 2  | )
676      0670 1  | END;
```

```

                                0004 00000 INV_COMMAND:
                                .WORD
                                Save R2
                                52 00000000' EF 9E 00002 MOVAB COMMAND_DESC+4, R2
                                7E 08 AC 7D 00009 MOVQ 8(TPARSE_BLOCK), -(SP)
                                7E 10 AC 7D 0000D MOVQ 16(TPARSE_BLOCK), -(SP)
                                2E 14 AC 62 DD 00011 PUSHL COMMAND_DESC+4
                                08 06 AC 62 C3 00013 SUBL3 COMMAND_DESC+4, 20(TPARSE_BLOCK), -(SP)
                                00450010 8F DD 0001C BLBC 6(TPARSE_BLOCK), 1$
                                00450008 06 11 00022 BRB 2$
                                00000000G 00 8F DD 00024 1$: PUSHL #4521992
                                07 FB 0002A 2$: CALLS #7, LIB$STOP
                                04 00031 RET
```

; Routine Size: 50 bytes, Routine Base: \$CODE\$ + 0105

```

677      0671 1  |
678      0672 1  |
679      0673 1  | Give invalid switch status
680      0674 1  |
681      0675 1  |
682      0676 1  | ROUTINE INV_SWITCH =
683      0677 1  |
684      0678 2  | BEGIN
685      0679 2  | TPARSE_ARGS;
686      0680 2  |
687      P 0681 2  | ERR_EXIT ((IF .TPARSE_BLOCK[TPASV_ambiguous]
```

```

: 0655
: 0669
:
:
: 0670
```



```
... 688      P 0682      2      THEN DSKQ$_AMB_QUAL
... 689      P 0683      2      ELSE DSKQ$_INV_QUAL),
... 690      P 0684      2      .TPARSE_BLOCK[TPASL_TOKENPTR] = .COMMAND_DESC[1],
... 691      P 0685      2      .COMMAND_DESC[1],
... 692      P 0686      2      .TPARSE_BLOCK[TPASL_TOKENCNT],
... 693      P 0687      2      .TPARSE_BLOCK[TPASL_TOKENPTR],
... 694      P 0688      2      .TPARSE_BLOCK[TPASL_STRINGCNT],
... 695      P 0689      2      .TPARSE_BLOCK[TPASL_STRINGPTR],
... 696      P 0690      2      )
... 697      P 0691      1      END;
```

```
                                0004 00000 INV_SWITCH:
                                .WORD      Save R2
                                52 00000000' EF 9E 00002      MOVAB      COMMAND_DESC+4, R2
                                7E          08 AC 7D 00009      MOVQ      8(TPARSE_BLOCK), -(SP)
                                7E          10 AC 7D 0000D      MOVQ      16(TPARSE_BLOCK), -(SP)
                                7E          14 AC          62 DD 00011      PUSHL     COMMAND_DESC+4
                                08          06 AC          62 C3 00013      SUBL3     COMMAND_DESC+4, 20(TPARSE_BLOCK), -(SP)
                                00450020 BF DD 0001C      BLBC      6(TPARSE_BLOCK), 1$
                                00450018 BF DD 00022      PUSHL     #4522016
                                00000000G 00 06 11 0002C      BRB      2$
                                00000000G 00 BF DD 00024 1$:      PUSHL     #4522008
                                07 FB 0002A 2$:      CALLS     #7, LIB$STOP
                                04 00031      RET
```

```
... 0676
... 0690
... 0691
```

: Routine Size: 50 bytes, Routine Base: \$CODE\$ + 0137

```
... 698      0692      1      |
... 699      0693      1      |
... 700      0694      1      | Save the HELP key descriptor in the key descriptor vector.
... 701      0695      1      |
... 702      0696      1      |
... 703      0697      1      ROUTINE SAVE_KEY =
... 704      0698      1      |
... 705      0699      2      BEGIN
... 706      0700      2      |
... 707      0701      2      IF .KEY_INDEX LEQ (MAX_KEYS - 2)      ! check for too many keys
... 708      0702      2      THEN
... 709      0703      2      BEGIN
... 710      0704      2      KEY_VECTOR[.KEY_INDEX] = .TPARSE_BLOCK[TPASL_TOKENCNT];
... 711      0705      2      KEY_VECTOR[.KEY_INDEX+1] = .TPARSE_BLOCK[TPASL_TOKENPTR];
... 712      0706      2      KEY_INDEX = .KEY_INDEX+2;      ! Increment KEY_INDEX
... 713      0707      2      END;
... 714      0708      1      |
... 715      0709      1      END;
```

```
                                0004 00000 SAVE_KEY:
                                .WORD      Save R2
```

: 0697

DISKQUOTA
V04-000

C 12
15-Sep-1984 23:38:38 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:19:46 [DISKQ.SRC]DISKQUOTA.B32;1

Page 34
(7)

| | | | | | | | |
|---------|-----------|----|-------|-------|-------|-----------------------------------|------|
| 52 | 00000000' | EF | 9E | 00002 | MOVAB | KEY_INDEX, R2 | |
| 50 | | 62 | D0 | 00009 | MOVL | KEY_INDEX, R0 | |
| 0C | | 50 | D1 | 0000C | CML | R0, #12 | 0701 |
| | | 0F | 14 | 0000F | BGTR | 18 | |
| C8 A240 | 2C | A2 | D0 | 00011 | MOVL | TPARSE_BLOCK+16, KEY_VECTOR[R0] | 0704 |
| CC A240 | 30 | A2 | D0 | 00017 | MOVL | TPARSE_BLOCK+20, KEY_VECTOR+4[R0] | 0705 |
| 62 | | 02 | C0 | 0001D | ADDL2 | #2, KEY_INDEX | 0706 |
| 50 | | 01 | D0 | 00020 | MOVL | #1, R0 | 0709 |
| | | 04 | 00023 | 18: | RET | | |

; Routine Size: 36 bytes, Routine Base: \$CODE\$ + 0169

```

717 0710 1 GLOBAL ROUTINE USE_DEFAULT : NOVALUE =
718 0711 1
719 0712 1 ++
720 0713 1
721 0714 1 Functional Description:
722 0715 1
723 0716 1 This routine causes a USE SYSSDISK: command to be executed, to
724 0717 1 set up the channel to the default disk. If it fails, no error
725 0718 1 messages are output and the channel is simply left unassigned.
726 0719 1
727 0720 1 Calling Sequence:
728 0721 1 standard
729 0722 1
730 0723 1 Input Parameters:
731 0724 1 none
732 0725 1
733 0726 1 Implicit Inputs:
734 0727 1 none
735 0728 1
736 0729 1 Output Parameters:
737 0730 1 none
738 0731 1
739 0732 1 Implicit Outputs:
740 0733 1 none
741 0734 1
742 0735 1 Routines Called:
743 0736 1 none
744 0737 1
745 0738 1 Routine Value:
746 0739 1 none
747 0740 1
748 0741 1 Signals:
749 0742 1 none
750 0743 1
751 0744 1 Side Effects:
752 0745 1 none
753 0746 1
754 0747 1 --
755 0748 1
756 0749 2 BEGIN
757 0750 2
758 0751 2 BUILTIN
759 0752 2 CALLG; : linkage to action routines is CALLG
760 0753 2
761 0754 2
762 0755 2 : Enable the local condition handler to swallow error signals. Then plug
763 0756 2 the TPARSE control block and call the USE action routine.
764 0757 2
765 0758 2
766 0759 2 ENABLE DEF_HANDLER;
767 0760 2
768 0761 2 TPARSE_BLOCK[TPASL_TOKENCNT] = %CHARCOUNT ('SYSSDISK:');
769 0762 2 TPARSE_BLOCK[TPASL_TOKENPTR] = UPLIT BYTE ('SYSSDISK:');
770 0763 2 CALLG TPARSE_BLOCK, ACT_USE);
771 0764 2
772 0765 1 END; : end of routine USE_DEFAULT

```

3A 4B 53 49 44 24 53 59 53 00110 P.AAF: .PSECT \$SPLITS,NOWRT,NOEXE,2
.ASCII \SYS\$DISK:\

00000000V 04 A2 00000000' EF 9E 00002
6D 0015 CF DE 00009
62 09 D0 0000E
00000000V EF F0 A2 FA 00019
04 00021
0000 00022 1B:
7E D4 00024
5E DD 00026
00000000V 7E 04 AC 7D 00028
EF 03 FB 0002C
04 00033

.PSECT \$CODE\$,NOWRT,2
.ENTRY USE DEFAULT, Save R2
MOVAB TPARSE_BLOCK+16, R2
MOVAL 1\$, (FP)
MOVL #9, TPARSE_BLOCK+16
MOVAB P.AAF, TPARSE_BLOCK+20
CALLG TPARSE_BLOCK, ACT_USE
RET
.WORD Save nothing
CLRL -(SP)
PUSHL SP
MOVQ 4(AP), -(SP)
CALLS #3, DEF_HANDLER
RET

0710
0749
0761
0762
0763
0765
0749

; Routine Size: 52 Lytes, Routine Base: \$CODE\$ + 018D


```
0774 0766 1 GLOBAL ROUTINE DEF_HANDLER (SIGNAL, MECHANISM) : NOVALUE =
0775 0767 1
0776 0768 1 **
0777 0769 1
0778 0770 1 Functional Description:
0779 0771 1
0780 0772 1 This routine is the condition handler for the preceding routine.
0781 0773 1 It simply unwinds the stack on any signal.
0782 0774 1
0783 0775 1 Calling Sequence:
0784 0776 1 standard
0785 0777 1
0786 0778 1 Input Parameters:
0787 0779 1 none
0788 0780 1
0789 0781 1 Implicit Inputs:
0790 0782 1 none
0791 0783 1
0792 0784 1 Output Parameters:
0793 0785 1 none
0794 0786 1
0795 0787 1 Implicit Outputs:
0796 0788 1 none
0797 0789 1
0798 0790 1 Routines Called:
0799 0791 1 none
0800 0792 1
0801 0793 1 Routine Value:
0802 0794 1 none
0803 0795 1
0804 0796 1 Signals:
0805 0797 1 none
0806 0798 1
0807 0799 1 Side Effects:
0808 0800 1 none
0809 0801 1
0810 0802 1 --
0811 0803 1
0812 0804 2 BEGIN
0813 0805 2
0814 0806 2 MAP
0815 0807 2 SIGNAL : REF BBLOCK, ! signal vector
0816 0808 2 MECHANISM : REF BBLOCK; ! mechanism vector
0817 0809 2
0818 0810 2
0819 0811 2 SUNWIND ();
0820 0812 2
0821 0813 1 END; ! end of routine DEF_HANDLER
```

.EXTRN SYSSUNWIND

00000000G 00

```
0000 00000
7E 7C 00002
02 FB 00004
04 00008
```

```
.ENTRY DEF_HANDLER, Save nothing
CLRQ -(SP)
CALLS #2, SYSSUNWIND
RET
```

```
: 0766
: 0811
: 0813
```

DISKQUOTA
V04-000

6 12
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46

VAX-11 B11ss-32 V4.0-742
[DISKQ.SRC]DISKQUOTA.B32;1

Page 38
(9)

; Routine Size: 12 bytes, Routine Base: \$CODE\$ + 01C1

```
0814 1 GLOBAL ROUTINE ACT_USE =
0815 1
0816 1 ++
0817 1
0818 1 Functional Description:
0819 1
0820 1     This action routine processes the USE command. It assigns a channel
0821 1     to the specified device string.
0822 1
0823 1 Calling Sequence:
0824 1     standard
0825 1
0826 1 Input Parameters:
0827 1     none
0828 1
0829 1 Implicit Inputs:
0830 1     none
0831 1
0832 1 Output Parameters:
0833 1     none
0834 1
0835 1 Implicit Outputs:
0836 1     none
0837 1
0838 1 Routines Called:
0839 1     none
0840 1
0841 1 Routine Value:
0842 1     none
0843 1
0844 1 Signals:
0845 1     none
0846 1
0847 1 Side Effects:
0848 1     none
0849 1
0850 1 --
0851 1
0852 2 BEGIN
0853 2
0854 2 LITERAL
0855 2     BUFFER_LEN      = 64;           ! string buffer length
0856 2
0857 2 LOCAL
0858 2     P               ! general string pointer
0859 2     STATUS          ! general status value
0860 2     NAME_DESC       : VECTOR [2],  ! descriptor of logical name to translate
0861 2     RESULT          : VECTOR [2],  ! descriptor of translated name
0862 2     STRING_BUFFER   : VECTOR [BUFFER_LEN, BYTE]; ! string buffer (obviously)
0863 2
0864 2 TPARSE_ARGS;           ! declare TPARSE argument list
0865 2
0866 2
0867 2 ! Get the device name string and attempt to do logical name translation.
0868 2 ! We iterate on logical name translation until the service returns $$$NOTRAN.
0869 2 ! Perform device name extraction by using only the part of the logical name to
0870 2 ! the left of the colon (if any), also checking for node names.
0871 2
0872 2
0873 2
0874 2
0875 2
0876 2
0877 2
0878 2
0879 2
```

```
880 0871 :
881 0872 :
882 0873 IF .CHANNEL NEQ 0
883 0874 THEN $DASSGN (CHAN = .CHANNEL);
884 0875 CHANNEL = 0;
885 0876 :
886 0877 RESULT[0] = BUFFER_LEN;
887 0878 RESULT[1] = STRING_BUFFER;
888 0879 NAME_DESC[0] = .TPARSE_BLOCK[TPASL_TOKENCNT]; ! get initial logical name
889 0880 NAME_DESC[1] = STRING_BUFFER;
890 0881 CHSCOPY (.TPARSE_BLOCK[TPASL_TOKENCNT], .TPARSE_BLOCK[TPASL_TOKENPTR], 0, .RESULT[0], .RESULT[1]);
891 0882 :
892 0883 IF BEGIN
893 0884   DECR N FROM 10 TO 1 DO
894 0885     BEGIN
895 0886       P = CHSFIND CH (.NAME_DESC[0], .NAME_DESC[1], ':');
896 0887       IF NOT CH$FAIL (.P)
897 0888         THEN
898 0889           BEGIN
899 0890             IF .P - .NAME_DESC[1] LSSU .NAME_DESC[0] - 1
900 0891             AND .(.P) < 0, 16 > EQL ';;'
901 0892             THEN ERR_EXIT (DSKQ$NONLOCAL);
902 0893             NAME_DESC[0] = .P - .NAME_DESC[1];
903 0894             END;
904 0895           IF CH$RCHAR (.NAME_DESC[1]) EQL '-'
905 0896             THEN EXITLOOP 0;
906 0897           STATUS = $TRNLOG (LOGNAM = NAME_DESC[0],
907 0898                             RSLLEN = NAME_DESC[0],
908 0899                             RSLBUF = RESULT[0]);
909 0900           IF .STATUS EQL SS$NOTRAN THEN EXITLOOP 0;
910 0901           IF NOT .STATUS THEN ERR_EXIT (.STATUS);
911 0902           END
912 0903         END
913 0904       THEN ERR_EXIT (DSKQ$NOTRAN);
914 0905       RESULT[0] = .NAME_DESC[0];
915 0906       ! Now assign a channel to the device name.
916 0907       !
917 0908       STATUS = $ASSIGN (DEVNAM = RESULT[0], CHAN = CHANNEL);
918 0909       IF NOT .STATUS
919 0910       THEN ERR_EXIT (.STATUS);
920 0911       1
921 0912       !
922 0913       !
923 0914       !
924 0915       !
925 0916       !
926 0917       !
927 0918       !
928 0919 END; ! end of routine ACT_USE
```

```
.EXTRN SYS$DASSGN, SYS$TRNLOG
.EXTRN SYS$ASSIGN
```

```
00FC 0000
57 00000000 EF 9E 00002
```

```
.ENTRY ACT_USE, Save R2,R3,R4,R5,R6,R7
MOVAB CHANNEL, R7
```

```
: 0814
:
```


| | | | | | | | | | | |
|----|----|-----------|-----------|----------|----|-------|--------|----------------|--|------|
| | | 56 | 00000000G | 00 | 9E | 00009 | MOVAB | LIB\$STOP, R6 | | |
| | | 5E | B0 | AE | 9E | 00010 | MOVAB | -80(SP), SP | | |
| | | 50 | | 67 | 3C | 00014 | MOVZWL | CHANNEL, R0 | | 0873 |
| | | | | 09 | 13 | 00017 | BEQL | 1\$ | | |
| | | | | 50 | DD | 00019 | PUSHL | R0 | | 0874 |
| | | 00000000G | 00 | 01 | FB | 0001B | CALLS | #1, SYSSDASSGN | | |
| | | | | 67 | B4 | 00022 | CLRW | CHANNEL | | 0875 |
| | | 40 | AE | 40 | BF | 9A | 00024 | MOVZBL | #64, RESULT | 0877 |
| | | 44 | AE | | 6E | 9E | 00029 | MOVAB | STRING BUFFER, RESULT+4 | 0878 |
| | | 48 | AE | 10 | AC | D0 | 0002D | MOVL | 16(TPARSE_BLOCK), NAME_DESC | 0879 |
| | | 4C | AE | | 6E | 9E | 00032 | MOVAB | STRING BUFFER, NAME_DESC+4 | 0880 |
| 40 | AE | DO | 14 | BC | AC | 2C | 00036 | MOVCS | 16(TPARSE_BLOCK), @20(TPARSE_BLOCK), #0, - | 0881 |
| | | | | 44 | BE | | 0003E | | RESULT, @RESULT+4 | |
| | | | | 54 | OA | D0 | 00040 | MOVL | #10, N | 0884 |
| | | 4C | BE | 48 | 3A | 3A | 00043 | LOCC | #58, NAME_DESC, @NAME_DESC+4 | 0886 |
| | | | | | 02 | 12 | 00049 | BNEQ | 3\$ | |
| | | | | 55 | 51 | D4 | 0004B | CLRL | R1 | |
| | | | | | 51 | D0 | 0004D | MOVL | R1, P | |
| | | | | 52 | 23 | 13 | 00050 | BEQL | 5\$ | 0887 |
| | | 52 | | 55 | AE | C3 | 00052 | SUBL3 | NAME_DESC+4, P, R2 | 0890 |
| | | 50 | | 48 | AE | C3 | 00057 | SUBL3 | #1, NAME_DESC, R0 | |
| | | | | | 50 | D1 | 0005C | CMPL | R2, R0 | |
| | | | | | 10 | 1E | 0005F | BGEQU | 4\$ | |
| | | 3A3A | 8F | | 65 | B1 | 00061 | CPW | (P), #14906 | 0891 |
| | | | | | 09 | 12 | 00066 | BNEQ | 4\$ | |
| | | | 00450038 | | 8F | DD | 00068 | PUSHL | #4522040 | 0892 |
| | | | | 66 | 01 | FB | 0006E | CALLS | #1, LIB\$STOP | |
| | | 48 | AE | | 52 | D0 | 00071 | MOVL | R2, NAME_DESC | 0893 |
| | | 5F | BF | 4C | BE | 91 | 00075 | CMPB | @NAME_DESC+4, #95 | 0896 |
| | | | | | 34 | 13 | 0007A | BEQL | 7\$ | |
| | | | | | 7E | 7C | 0007C | CLRQ | -(SP) | 0901 |
| | | | | | 7E | D4 | 0007E | CLRL | -(SP) | |
| | | | | 4C | AE | 9F | 00080 | PUSHAB | RESULT | |
| | | | | 58 | AE | 9F | 00083 | PUSHAB | NAME_DESC | |
| | | | | 5C | AE | 9F | 00086 | PUSHAB | NAME_DESC | |
| | | 00000000G | 00 | | 06 | FB | 00089 | CALLS | #6, SYS\$TRNLOG | |
| | | | 53 | | 50 | D0 | 00090 | MOVL | R0, STATUS | |
| | | 00000629 | 8F | | 53 | D1 | 00093 | CMPL | STATUS, #1577 | 0902 |
| | | | | | 14 | 13 | 0009A | BEQL | 7\$ | |
| | | | 05 | | 53 | E8 | 0009C | BLBS | STATUS, 6\$ | 0903 |
| | | | | | 53 | DD | 0009F | PUSHL | STATUS | |
| | | | 66 | | 01 | FB | 000A1 | CALLS | #1, LIB\$STOP | |
| | | | 9C | | 54 | F5 | 000A4 | SOBGTR | N, 2\$ | 0884 |
| | | | | 00450C40 | 8F | DD | 000A7 | PUSHL | #4522048 | 0906 |
| | | | 66 | | 01 | FB | 000AD | CALLS | #1, LIB\$STOP | |
| | | 40 | AE | 48 | AE | D0 | 000B0 | MOVL | NAME_DESC, RESULT | 0908 |
| | | | | | 7E | 7C | 000B5 | CLRQ | -(SP) | 0913 |
| | | | | | 57 | DD | 000B7 | PUSHL | R7 | |
| | | | | 4C | AE | 9F | 000B9 | PUSHAB | RESULT | |
| | | 00000000G | 00 | | 04 | FB | 000BC | CALLS | #4, SYSS\$ASSIGN | |
| | | | 53 | | 50 | D0 | 000C3 | MOVL | R0, STATUS | |
| | | | 05 | | 53 | E8 | 000C6 | BLBS | STATUS, 8\$ | 0914 |
| | | | | | 53 | DD | 000C9 | PUSHL | STATUS | 0915 |
| | | | 66 | | 01 | FB | 000CB | CALLS | #1, LIB\$STOP | |
| | | | 50 | | 01 | D0 | | | | |

DISKQUOTA
V04-000

5 12
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46

VAX-11 Bliss-32 V4.0-742
[DISK0.SRC]DISKQUOTA.B32;1

Page 42
(10)

; Routine Size: 210 bytes, Routine Base: \$CODES + 01CD

D1
VO

```
0930 1 GLOBAL ROUTINE ACT_CREATE =
0931 1
0932 1 **
0933 1
0934 1 Functional Description:
0935 1
0936 1 This action routine implements the CREATE command. It creates the
0937 1 disk quota file and activates it.
0938 1
0939 1 Calling Sequence:
0940 1 standard
0941 1
0942 1 Input Parameters:
0943 1 none
0944 1
0945 1 Implicit Inputs:
0946 1 none
0947 1
0948 1 Output Parameters:
0949 1 none
0950 1
0951 1 Implicit Outputs:
0952 1 none
0953 1
0954 1 Routines Called:
0955 1 none
0956 1
0957 1 Routine Value:
0958 1 none
0959 1
0960 1 Signals:
0961 1 none
0962 1
0963 1 Side Effects:
0964 1 none
0965 1
0966 1 --
0967 1
0968 2 BEGIN
0969 2
0970 2 BIND
0971 2 QFILE_DATA = UPLIT (1, 0, 0, 1000, 100, REP 123 OF (0));
0972 2
0973 2 PSECT
0974 2 PLIT = $OWNS;
0975 2
0976 2 BIND
0977 2 CREATE_ATTRIB = UPLIT (WORD (FATSC_LENGTH, ATRSC_RECATTR),
0978 2 UPLIT (BYTE (FATSC_FIXED, 0), WORD (DQFSC_LENGTH,
0979 2 1*16, 2*16, WORD (0, 0, DQFSC_LENGTH, 0)),
0980 2 0);
0981 2
0982 2 PSECT
0983 2 PLIT = $PLITS;
0984 2
0985 2 LOCAL
0986 2 STATUS: ! general status value
```

```

987 0977
988 0978
989 0979 : Verify that a channel is open.
990 0980
991 0981
992 0982 IF .CHANNEL EQL 0
993 0983 THEN ERR_EXIT (DSKQ$NODEVICE);
994 0984
995 0985 : Create the quota file.
996 0986
997 0987
998 0988 QUOTA_FIB[FIB$W_DID_NUM] = FIB$C_MFD;
999 0989 QUOTA_FIB[FIB$W_DID_SEQ] = FIB$C_MFD;
1000 0990 QUOTA_FIB[FIB$W_DID_RVN] = 1;
1001 0991 QUOTA_FIB[FIB$L_ACCTL] = FIB$M_WRITE OR FIB$M_NOREAD;
1002 0992 QUOTA_FIB[FIB$W_EXCTL] = FIB$M_EXTEND OR FIB$M_ALCON OR FIB$M_FILCON;
1003 0993 QUOTA_FIB[FIB$L_EXSZ] = 1;
1004 0994 QUOTA_FIB[FIB$B_ALALIGN] = FIB$C_LBN;
1005 0995 QUOTA_FIB[FIB$W_LOC_RVN] = 1;
1006 0996
1007 0997 STATUS = $QIOW (CHAN = .CHANNEL,
1008 0998             FUNC = IO$CREATE OR IO$M_CREATE OR IO$M_ACCESS,
1009 0999             IOSB = IO_STATUS,
1010 1000             P1 = QFIB_DESC,
1011 1001             P2 = QFILE_NAME,
1012 1002             P3 = CREATE_ATTRIB
1013 1003             );
1014 1004 IF .STATUS THEN STATUS = .IO_STATUS[0];
1015 1005 IF NOT .STATUS
1016 1006 THEN ERR_EXIT (DSKQ$CREATERR, .STATUS);
1017 1007
1018 1008 : Write the initial data block and close the file.
1019 1009
1020 1010
1021 1011 STATUS = $QIOW (CHAN = .CHANNEL,
1022 1012             FUNC = IO$WRITEVBLK,
1023 1013             IOSB = IO_STATUS,
1024 1014             P1 = QFILE_DATA,
1025 1015             P2 = 512,
1026 1016             P3 = 1
1027 1017             );
1028 1018 IF .STATUS THEN STATUS = .IO_STATUS[0];
1029 1019 IF NOT .STATUS
1030 1020 THEN ERR_EXIT (DSKQ$INITERR, .STATUS);
1031 1021
1032 1022 STATUS = $QIOW (CHAN = .CHANNEL,
1033 1023             FUNC = IO$DEACCESS,
1034 1024             IOSB = IO_STATUS
1035 1025             );
1036 1026 IF .STATUS THEN STATUS = .IO_STATUS[0];
1037 1027 IF NOT .STATUS
1038 1028 THEN ERR_EXIT (DSKQ$CLOSERR, .STATUS);
1039 1029
1040 1030 : Now activate the quota file.
1041 1031
1042 1032
1043 1033 QUOTA_FIB[FIB$W_DID_NUM] = 0;
```



```
1044 1034 2 QUOTA_FIB[FIBSW_DID_SEQ] = 0;
1045 1035 2 QUOTA_FIB[FIBSW_DID_RVN] = 0;
1046 1036 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_ENA_QUOTA;
1047 1037 2 QUOTA_FIB[FIBSL_CNTRLVAL] = 0;
1048 1038 2 STATUS = $QIOW TCHAN = CHANNEL,
1049 1039 2 FUNC = IOS.ACPCONTROL,
1050 1040 2 IOSB = IO STATUS,
1051 1041 2 P1 = QFIB_DESC
1052 1042 2 );
1053 1043 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1054 1044 2 IF NOT .STATUS
1055 1045 2 THEN ERR_EXIT (DSKQS_ACTERR, .STATUS);
1056 1046 2
1057 1047 2 1
1058 1048 2 END;
```

! end of routine ACT_CREATE

```
00000064 000003E8 00000000 00000000 00000001 00119
00000000# 0011C P.AAG: .BLKB 3
00000000# 00130 .LONG 1, 0, 0, 1000, 100
00000000# 00130 .LONG 0[123]
```

```
0000 0020 0000 0000 0004 0020 00000000# 00264
00000000# 00268 .PSECT $OWNS,NOEXE,2
P.AAI: .BYTE 1, 0
0024C 0024E .WORD 32
00250 .LONG 65536, 131072
P.AAH: .WORD 0, 0, 32, 0
00260 .WORD 32, 4
00264 .ADDRESS P.AAI
00268 .LONG 0
```

```
QFILE DATA= P.AAG
CREATE_ATTRIB= P.AAH
.EXTRN SYSSQIOW
```

```
003C 00000 .PSECT $CODES,NOVRT,2
55 00000000G 00 9E 00002 .ENTRY ACT_CREATE, Save R2,R3,R4,R5
54 00000000G 00 9E 00009 MOVAB SYSSQIOW, R5
53 00000000' EF 9E 00010 MOVAB LIB$STOP, R4
FC A3 B5 00017 MOVAB IO STATUS, R3
09 12 0001A TSTW CHANNEL
0F DD 0001C BNEQ 18
00450048 8F DD 0001C PUSHL #4522056
64 C3 00040004 01 FB 00022 CALLS #1, LIB$STOP
0182 C3 0500 8F DD 00025 18: MOVL #262148, QUOTA_FIB+10
0186 C3 01 B0 0002E MOVW #1, QUOTA_FIB+T4
0178 C3 8F 3C 00033 MOVZWL #1280, QUOTA_FIB
018E C3 8F 9B 0003A MOVZBW #133, QUOTA_FIB+22
0190 C3 01 D0 00040 MOVL #1, QUOTA_FIB+24
0199 C3 02 90 00045 MOVW #2, QUOTA_FIB+33
019E C3 01 B0 0004A MOVW #1, QUOTA_FIB+38
025C C3 7E D4 0004F CLRL -(SP)
00051 9F 00051 PUSHAB CREATE_ATTRIB
```

| | | | | | | | |
|------|----------|----|----|-------|--------|-------------------|------|
| | | 7E | 7C | 00055 | CLRQ | -(SP) | |
| | 0240 | C3 | 9F | 00057 | PUSHAB | QFILE_NAME | |
| | 0208 | C3 | 9F | 00058 | PUSHAB | QFIB_DESC | |
| | | 7E | 7C | 0005F | CLRQ | -(SPT) | |
| | | 53 | DD | 00061 | PUSHL | R3 | |
| 7E | F3 | 8F | 9A | 00063 | MOVZBL | #243, -(SP) | |
| 7E | FC | A3 | 3C | 00067 | MOVZWL | CHANNEL, -(SP) | |
| | | 7E | D4 | 00068 | CLRL | -(SP) | |
| 65 | | 0C | FB | 0006D | CALLS | #12, SYSSQIOW | |
| 52 | | 50 | DD | 00070 | MOVL | R0, STATUS | |
| 06 | | 52 | E9 | 00073 | BLBC | STATUS, 2\$ | 1004 |
| 52 | | 63 | 3C | 00076 | MOVZWL | IO STATUS, STATUS | |
| 0B | | 52 | E8 | 00079 | BLBS | STATUS, 3\$ | 1005 |
| | | 52 | DD | 0007C | PUSHL | STATUS | 1006 |
| | 00450050 | 8F | DD | 0007E | PUSHL | #4522064 | |
| 64 | | 02 | FB | 00084 | CALLS | #2, LIB\$STOP | |
| | | 7E | 7C | 00087 | CLRQ | -(SP) | 1017 |
| 7E | | 01 | 7D | 00089 | MOVQ | #1, -(SP) | |
| 7E | 0200 | 8F | 3C | 0008C | MOVZWL | #512, -(SP) | |
| | 00000000 | EF | 9F | 00091 | PUSHAB | QFILE_DATA | |
| | | 7E | 7C | 00097 | CLRQ | -(SP) | |
| | | 53 | DD | 00099 | PUSHL | R3 | |
| | | 30 | DD | 0009B | PUSHL | #48 | |
| 7E | FC | A3 | 3C | 0009D | MOVZWL | CHANNEL, -(SP) | |
| | | 7E | D4 | 000A1 | CLRL | -(SP) | |
| 65 | | 0C | FB | 000A3 | CALLS | #12, SYSSQIOW | |
| 52 | | 50 | DD | 000A6 | MOVL | R0, STATUS | |
| 06 | | 52 | E9 | 000A9 | BLBC | STATUS, 4\$ | 1018 |
| 52 | | 63 | 3C | 000AC | MOVZWL | IO STATUS, STATUS | |
| 0B | | 52 | E8 | 000AF | BLBS | STATUS, 5\$ | 1019 |
| | | 52 | DD | 000B2 | PUSHL | STATUS | 1020 |
| | 00450058 | 8F | DD | 000B4 | PUSHL | #4522072 | |
| 64 | | 02 | FB | 000BA | CALLS | #2, LIB\$STOP | |
| | | 7E | 7C | 000BD | CLRQ | -(SP) | 1025 |
| | | 7E | 7C | 000BF | CLRQ | -(SP) | |
| | | 7E | 7C | 000C1 | CLRQ | -(SP) | |
| | | 7E | 7C | 000C3 | CLRQ | -(SP) | |
| | | 53 | DD | 000C5 | PUSHL | R3 | |
| | | 34 | DD | 000C7 | PUSHL | #52 | |
| 7E | FC | A3 | 3C | 000C9 | MOVZWL | CHANNEL, -(SP) | |
| | | 7E | D4 | 000CD | CLRL | -(SP) | |
| 65 | | 0C | FB | 000CF | CALLS | #12, SYSSQIOW | |
| 52 | | 50 | DD | 000D2 | MOVL | R0, STATUS | |
| 06 | | 52 | E9 | 000D5 | BLBC | STATUS, 6\$ | 1026 |
| 52 | | 63 | 3C | 000D8 | MOVZWL | IO STATUS, STATUS | |
| 0B | | 52 | E8 | 000DB | BLBS | STATUS, 7\$ | 1027 |
| | | 52 | DD | 000DE | PUSHL | STATUS | 1028 |
| | 00450060 | 8F | DD | 000E0 | PUSHL | #4522080 | |
| 64 | | 02 | FB | 000E6 | CALLS | #2, LIB\$STOP | |
| | 0182 | C3 | D4 | 000E9 | CLRL | QUOTA_FIB+10 | 1033 |
| | 0186 | C3 | B4 | 000ED | CLRW | QUOTA_FIB+14 | 1035 |
| 018E | | 09 | B0 | 000F1 | MOVW | #9, QUOTA_FIB+22 | 1036 |
| | 0190 | C3 | D4 | 000F6 | CLRL | QUOTA_FIB+24 | 1037 |
| | | 7E | 7C | 000FA | CLRQ | -(SP) | 1042 |
| | | 7E | 7C | 000FC | CLRQ | -(SP) | |
| | | 7E | D4 | 000FE | CLRL | -(SP) | |
| | 0208 | C3 | 9F | 00100 | PUSHAB | QFIB_DESC | |

DIS
V04

1043
1044
1045
1048

; Routine Size: 302 bytes, Routine Base: \$CODES + 029F

```
1060 1049 1 GLOBAL ROUTINE ACT_ENABLE =
1061 1050 1
1062 1051 1 ++
1063 1052 1
1064 1053 1 Functional Description:
1065 1054 1
1066 1055 1 This action routine implements the ENABLE command. It enables the
1067 1056 1 disk quota file.
1068 1057 1
1069 1058 1 Calling Sequence:
1070 1059 1 standard
1071 1060 1
1072 1061 1 Input Parameters:
1073 1062 1 none
1074 1063 1
1075 1064 1 Implicit Inputs:
1076 1065 1 none
1077 1066 1
1078 1067 1 Output Parameters:
1079 1068 1 none
1080 1069 1
1081 1070 1 Implicit Outputs:
1082 1071 1 none
1083 1072 1
1084 1073 1 Routines Called:
1085 1074 1 none
1086 1075 1
1087 1076 1 Routine Value:
1088 1077 1 none
1089 1078 1
1090 1079 1 Signals:
1091 1080 1 none
1092 1081 1
1093 1082 1 Side Effects:
1094 1083 1 none
1095 1084 1
1096 1085 1 --
1097 1086 1
1098 1087 2 BEGIN
1099 1088 2
1100 1089 2
1101 1090 2 LOCAL
1102 1091 2 STATUS;
1103 1092 2 ! general status value
1104 1093 2
1105 1094 2 ! Verify that a channel is open.
1106 1095 2
1107 1096 2
1108 1097 2 IF .CHANNEL EQL 0
1109 1098 2 THEN ERR_EXIT (DSKQ$_NODEVICE);
1110 1099 2
1111 1100 2 ! Now activate the quota file.
1112 1101 2
1113 1102 2
1114 1103 2 QUOTA_FIB[FIB$W_DID_NUM] = FIDSC_MFD;
1115 1104 2 QUOTA_FIB[FIB$W_DID_SEQ] = FIDSC_MFD;
1116 1105 2 QUOTA_FIB[FIB$W_DID_RVN] = 1;
```



```
1117 1106 2 QUOTA_FIB[FIB$W CNTRLFUNC] = FIB$C_ENA_QUOTA;
1118 1107 STATUS = $QIOW TCHAN = CHANNEL;
1119 1108 FUNC = IO$ACPCONTROL;
1120 1109 IOSB = IO STATUS;
1121 1110 P1 = QFIB_DESC;
1122 1111 P2 = QFILE_NAME;
1123 1112 );
1124 1113 IF .STATUS THEN STATUS = .IO_STATUS[0];
1125 1114 IF NOT .STATUS
1126 1115 THEN ERR_EXIT (DSKQ$ACTERR, .STATUS);
1127 1116
1128 1117 1
1129 1118 1 END;

! end of routine ACT_ENABLE
```

| | | | | | | |
|-----------|----|-----------|-------------|--------|------------------------|------|
| | | | 000C 00000 | .ENTRY | ACT_ENABLE, Save R2,R3 | 1049 |
| | 53 | 00000000G | 00 9E 00002 | MOVAB | LIB\$STOP, R3 | |
| | 52 | 000000000 | EF 9E 00009 | MOVAB | CHANNEL, R2 | |
| | | | 62 B5 00010 | TSTW | CHANNEL | 1097 |
| | | | 09 12 00012 | BNEQ | 1\$ | |
| | | 00450048 | 8F DD 00014 | PUSHL | #4522056 | 1098 |
| | 63 | | 01 FB 0001A | CALLS | #1, LIB\$STOP | |
| 0186 | C2 | 00040004 | 8F D0 0001D | MOVL | #262148, QUOTA_FIB+10 | 1103 |
| 018A | C2 | | 01 B0 00026 | MOVW | #1, QUOTA_FIB+14 | 1105 |
| 0192 | C2 | | 09 B0 0002B | MOVW | #9, QUOTA_FIB+22 | 1106 |
| | | | 7E 7C 00030 | CLRQ | -(SP) | 1112 |
| | | | 7E 7C 00032 | CLRQ | -(SP) | |
| | | 0244 | C2 9F 00034 | PUSHAB | QFILE_NAME | |
| | | 020C | C2 9F 00038 | PUSHAB | QFIB_DESC | |
| | | | 7E 7C 0003C | CLRQ | -(SP) | |
| | | 04 | A2 9F 0003E | PUSHAB | IO STATUS | |
| | | | 38 DD 00041 | PUSHL | #58 | |
| | 7E | | 62 3C 00043 | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E D4 00046 | CLRL | -(SP) | |
| 00000000G | 00 | | 0C FB 00048 | CALLS | #12, SYS\$QIOW | |
| | 07 | | 50 E9 0004F | BLBC | STATUS, 2\$ | 1113 |
| | 50 | 04 | A2 3C 00052 | MOVZWL | IO STATUS, STATUS | |
| | 0B | | 50 E8 00056 | BLBS | STATUS, 3\$ | 1114 |
| | | | 50 DD 00059 | PUSHL | STATUS | 1115 |
| | | 00450068 | 8F DD 0005B | PUSHL | #4522088 | |
| | 63 | | 02 FB 00061 | CALLS | #2, LIB\$STOP | |
| | 50 | | 01 D0 00064 | MOVL | #1, R0 | 1118 |
| | | | 04 00067 | RET | | |

; Routine Size: 104 bytes, Routine Base: \$CODE\$ + 03CD

```
1131 1119 1 GLOBAL ROUTINE ACT_DISABLE =
1132 1120 1
1133 1121 1 ++
1134 1122 1
1135 1123 1 Functional Description:
1136 1124 1
1137 1125 1 This action routine implements the DISABLE command. It disables the
1138 1126 1 disk quota file.
1139 1127 1
1140 1128 1 Calling Sequence:
1141 1129 1 standard
1142 1130 1
1143 1131 1 Input Parameters:
1144 1132 1 none
1145 1133 1
1146 1134 1 Implicit Inputs:
1147 1135 1 none
1148 1136 1
1149 1137 1 Output Parameters:
1150 1138 1 none
1151 1139 1
1152 1140 1 Implicit Outputs:
1153 1141 1 none
1154 1142 1
1155 1143 1 Routines Called:
1156 1144 1 none
1157 1145 1
1158 1146 1 Routine Value:
1159 1147 1 none
1160 1148 1
1161 1149 1 Signals:
1162 1150 1 none
1163 1151 1
1164 1152 1 Side Effects:
1165 1153 1 none
1166 1154 1
1167 1155 1 --
1168 1156 1
1169 1157 2 BEGIN
1170 1158 2
1171 1159 2
1172 1160 2 LOCAL
1173 1161 2 STATUS; ! general status value
1174 1162 2
1175 1163 2
1176 1164 2 ! Verify that a channel is open.
1177 1165 2 !
1178 1166 2
1179 1167 2 IF .CHANNEL EQL 0
1180 1168 2 THEN ERR_EXIT (DSKQ$NODEVICE);
1181 1169 2
1182 1170 2 ! Now deactivate the quota file.
1183 1171 2 !
1184 1172 2
1185 1173 2 QUOTA_FIB[FIB$W_CNTRLFUNC] = FIB$C_DSA_QUOTA;
1186 1174 2 QUOTA_FIB[FIB$L_CNTRLVAL] = 0;
P 1187 1175 2 STATUS = $QIOW TCHAN = .CHANNEL,
```

```

: 1188
: 1189
: 1190
: 1191
: 1192
: 1193
: 1194
: 1195
: 1196
: 1197

P 1176 2
P 1177 2
P 1178 2
P 1179 2
1180 IF .STATUS THEN STATUS = .IO_STATUS[0];
1181 IF NOT .STATUS
1182 THEN ERR_EXIT (DSKQ$_DACTERR, .STATUS);
1183
1184 1
1185 1 END;
```

! end of routine ACT_DISABLE

| | | | | | | |
|-----------|-----------|-------------|--------|-------------------|------------|------|
| | | 000C 00000 | .ENTRY | ACT_DISABLE | Save R2,R3 | 1119 |
| 53 | 00000000G | 00 9E 00002 | MOVAB | LIB\$STOP, R3 | | |
| 52 | 000000000 | EF 9E 00009 | MOVAB | CHANNEL, R2 | | |
| | | 62 B5 00010 | TSTW | CHANNEL | | 1167 |
| | | 09 12 00012 | BNEQ | 1\$ | | |
| | 00450048 | 8F DD 00014 | PUSHL | #4522056 | | 1168 |
| 0192 | 63 | 01 FB 0001A | CALLS | #1, LIB\$STOP | | |
| | C2 | 0A B0 0001D | MOVW | #10, QUOTA_FIB+22 | | 1173 |
| | 0194 | C2 D4 00022 | CLRL | QUOTA_FIB+24 | | 1174 |
| | | 7E 7C 00026 | CLRQ | -(SP) | | 1179 |
| | | 7E 7C 00028 | CLRQ | -(SP) | | |
| | | 7E D4 0002A | CLRL | -(SP) | | |
| | 020C | C2 9F 0002C | PUSHAB | QFIB_DESC | | |
| | 04 | 7E 7C 00030 | CLRQ | -(SP) | | |
| | | A2 9F 00032 | PUSHAB | IO_STATUS | | |
| | 7E | 38 DD 00035 | PUSHL | #56 | | |
| | | 62 3C 00037 | MOVZWL | CHANNEL, -(SP) | | |
| 00000000G | 00 | 7E D4 0003A | CLRL | -(SP) | | |
| | 07 | 0C FB 0003C | CALLS | #12, SYS\$QIOW | | 1180 |
| | 50 | 50 E9 00043 | BLBC | STATUS, 2\$ | | |
| | 04 | A2 3C 00046 | MOVZWL | IO_STATUS, STATUS | | 1181 |
| | 0B | 50 E8 0004A | BLBS | STATUS, 3\$ | | 1182 |
| | | 50 DD 0004D | PUSHL | STATUS | | |
| | 00450070 | 8F DD 0004F | PUSHL | #4522096 | | |
| | 63 | 02 FB 00055 | CALLS | #2, LIB\$STOP | | |
| | 50 | 01 D0 00058 | MOVL | #1, R0 | | 1185 |
| | | 04 0005B | RET | | | |

; Routine Size: 92 bytes, Routine Base: \$CODE\$ + 0435

```
1199 1186 1 GLOBAL ROUTINE ACT_ADD =
1200 1187 1
1201 1188 1 ++
1202 1189 1
1203 1190 1 Functional Description:
1204 1191 1
1205 1192 1 This action routine implements the ADD command. It adds the
1206 1193 1 specified entry to the quota file.
1207 1194 1
1208 1195 1 Calling Sequence:
1209 1196 1 standard
1210 1197 1
1211 1198 1 Input Parameters:
1212 1199 1 none
1213 1200 1
1214 1201 1 Implicit Inputs:
1215 1202 1 none
1216 1203 1
1217 1204 1 Output Parameters:
1218 1205 1 none
1219 1206 1
1220 1207 1 Implicit Outputs:
1221 1208 1 none
1222 1209 1
1223 1210 1 Routines Called:
1224 1211 1 none
1225 1212 1
1226 1213 1 Routine Value:
1227 1214 1 none
1228 1215 1
1229 1216 1 Signals:
1230 1217 1 none
1231 1218 1
1232 1219 1 Side Effects:
1233 1220 1 none
1234 1221 1
1235 1222 1 --
1236 1223 1
1237 1224 2 BEGIN
1238 1225 2
1239 1226 2
1240 1227 2 LOCAL
1241 1228 2 STATUS; ! general status value
1242 1229 2
1243 1230 2
1244 1231 2 ! Verify that a channel is open.
1245 1232 2 !
1246 1233 2
1247 1234 2 IF .CHANNEL EQL 0
1248 1235 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1249 1236 2
1250 1237 2 ! Validate the UIC to insure that there are no wildcards.
1251 1238 2 !
1252 1239 2
1253 1240 2 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP
1254 1241 2 OR .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER
1255 1242 2 THEN ERR_EXIT (DSKQS_INV_UIC);
```



```
1256 1243 2
1257 1244 2
1258 1245 2
1259 1246 2
1260 1247 2
1261 1248 2
1262 1249 2
1263 1250 2
1264 1251 2
1265 1252 2
1266 1253 2
1267 1254 2
1268 1255 2
1269 1256 2
1270 1257 2
1271 1258 2
1272 1259 2
1273 1260 2
1274 1261 2
1275 1262 2
1276 1263 2
1277 1264 2
1278 1265 2
1279 1266 2
1280 1267 2
1281 1268 2
1282 1269 2
1283 1270 2
1284 1271 2
1285 1272 2
1286 1273 2
1287 1274 2
1288 1275 2
1289 1276 2
1290 1277 2
1291 1278 2
1292 1279 2
1293 1280 2
1294 1281 2
1295 1282 2
1296 1283 2
1297 1284 2
1298 1285 2

! If either value is not specified, read the default record and copy its
! values into the unspecified fields.

IF NOT .UIC_FLAGS[PERM_SPEC]
OR NOT .UIC_FLAGS[OVER_SPEC]
THEN
  BEGIN
    QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_EXA_QUOTA;
    QUOTA_FIB[FIBSL_CNTRLVAL] = 0;
    QUOTA_FIB[FIBSL_WCC] = 0;
    STATUS = $QIOW (CHAN = .CHANNEL,
                    FUNC = IOS_ACPCONTROL,
                    IOSB = IO_STATUS,
                    P1 = QFIB_DESC,
                    P2 = DSTREC_DESC,
                    P4 = DSTREC_DESC
                   );
    IF NOT .UIC_FLAGS[PERM_SPEC]
    THEN PERM_VALUE = .DST_REC[DQFSL_PERMQUOTA];
    IF NOT .UIC_FLAGS[OVER_SPEC]
    THEN OVER_VALUE = .DST_REC[DQFSL_OVERDRAFT];
  END;

! Issue the ADD function call.

QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_ADD_QUOTA;
QUOTA_FIB[FIBSL_CNTRLVAL] = 0;
QUOTA_FIB[FIBSL_WCC] = 0;
STATUS = $QIOW (CHAN = .CHANNEL,
                FUNC = IOS_ACPCONTROL,
                IOSB = IO_STATUS,
                P1 = QFIB_DESC,
                P2 = SRCREC_DESC
               );
IF .STATUS THEN STATUS = .IO_STATUS[0];
IF NOT .STATUS
THEN ERR_EXIT (DSKQ$_ADDERR, .STATUS);

1
END;
```

! end of routine ACT_ADD

| | | | | | | | |
|------|-----------|----|----|-------|--------|------------------------|------|
| 54 | 00000000G | 00 | 9E | 00002 | .ENTRY | ACT_ADD, Save R2,R3,R4 | 1186 |
| 53 | 00000000G | 00 | 9E | 00009 | MOVAB | SYS\$QIOW, R4 | |
| 52 | 00000000' | EF | 9E | 00010 | MOVAB | LIB\$STOP, R3 | |
| | FE44 | C2 | B5 | 00017 | MOVAB | UIC_FLAGS, R2 | |
| | | 09 | 12 | 0001B | TST | CHANNEL | 1234 |
| | 00450048 | 8F | DD | 0001D | BNEQ | 1\$ | |
| 63 | | 01 | FB | 00023 | PUSHL | #4522056 | 1235 |
| 3FFF | 8F | A2 | B1 | 00026 | CALLS | #1, LIB\$STOP | |
| | | | | 1\$: | CMPW | UIC_VALUE+2, #16383 | 1240 |

| | | | | | | | | | | |
|----|------|----|----------|----|-------|-------|--------|------------------------|------|--|
| | FFFF | 8F | 84 | 08 | 13 | 0002C | BEQL | 28 | | |
| | | | | A2 | B1 | 0002E | CMPL | UIC_VALUE, #65535 | 1241 | |
| | | | 00450028 | 09 | 12 | 00034 | BNEQ | 38 | | |
| | | 63 | | 8F | DD | 00036 | PUSHL | #4522024 | 1242 | |
| 04 | | 62 | | 01 | FB | 0003C | CALLS | #1, LIB\$STOP | | |
| 38 | | 62 | | 03 | E1 | 0003F | BBC | #3, UIC_FLAGS, 48 | 1248 | |
| | | 62 | | 04 | E0 | 00043 | BBS | #4, UIC_FLAGS, 68 | 1249 | |
| | D6 | A2 | | 0C | B0 | 00047 | MOVW | #12, QUOTA_FIB+22 | 1252 | |
| | | | D8 | A2 | D4 | 0004B | CLRL | QUOTA_FIB+24 | 1253 | |
| | | | D0 | A2 | D4 | 0004E | CLRL | QUOTA_FIB+16 | 1254 | |
| | | | | 7E | 7C | 00051 | CLRL | -(SP) | 1261 | |
| | | | 48 | A2 | 9F | 00053 | PUSHAB | DSTREC_DESC | | |
| | | | | 7E | D4 | 00056 | CLRL | -(SP) | | |
| | | | 48 | A2 | 9F | 00058 | PUSHAB | DSTREC_DESC | | |
| | | | 50 | A2 | 9F | 0005B | PUSHAB | QFIB_DESC | | |
| | | | | 7E | 7C | 0005E | CLRL | -(SP) | | |
| | | | FE48 | C2 | 9F | 00060 | PUSHAB | IO STATUS | | |
| | | | | 38 | DD | 00064 | PUSHL | #58 | | |
| | | 7E | FE44 | C2 | 3C | 00066 | MOVZWL | CHANNEL, -(SP) | | |
| | | | | 7E | D4 | 0006B | CLRL | -(SP) | | |
| 05 | | 64 | | 0C | FB | 0006D | CALLS | #12, SYSSQIOW | | |
| | | 62 | | 03 | E0 | 00070 | BBS | #3, UIC_FLAGS, 58 | 1262 | |
| 05 | 8C | A2 | AC | A2 | D0 | 00074 | MOVL | DST_REC+12, PERM VALUE | 1263 | |
| | | 62 | | 04 | E0 | 00079 | BBS | #4, UIC_FLAGS, 68 | 1264 | |
| | 90 | A2 | B0 | A2 | D0 | 0007D | MOVL | DST_REC+16, OVER VALUE | 1265 | |
| | D6 | A2 | | 0B | B0 | 00082 | MOVW | #11, QUOTA_FIB+22 | 1271 | |
| | | | | D8 | A2 | 00086 | CLRL | QUOTA_FIB+24 | 1272 | |
| | | | | D0 | A2 | 00089 | CLRL | QUOTA_FIB+16 | 1273 | |
| | | | | 7E | 7C | 0008C | CLRL | -(SP) | 1279 | |
| | | | | 7E | 7C | 0008E | CLRL | -(SP) | | |
| | | | 40 | A2 | 9F | 00090 | PUSHAB | SRCREC_DESC | | |
| | | | 50 | A2 | 9F | 00093 | PUSHAB | QFIB_DESC | | |
| | | | | 7E | 7C | 00096 | CLRL | -(SP) | | |
| | | | FE48 | C2 | 9F | 00098 | PUSHAB | IO STATUS | | |
| | | | | 38 | DD | 0009C | PUSHL | #58 | | |
| | | 7E | FE44 | C2 | 3C | 0009E | MOVZWL | CHANNEL, -(SP) | | |
| | | | | 7E | D4 | 000A3 | CLRL | -(SP) | | |
| | | 64 | | 0C | FB | 000A5 | CALLS | #12, SYSSQIOW | | |
| | | 08 | | 50 | E9 | 000AB | BLBC | STATUS, 78 | 1280 | |
| | | 50 | FE48 | C2 | 3C | 000AB | MOVZWL | IO STATUS, STATUS | | |
| | | 08 | | 50 | E8 | 000B0 | BLBS | STATUS, 88 | 1281 | |
| | | | | 50 | DD | 000B3 | PUSHL | STATUS | 1282 | |
| | | | 00450078 | 8F | DD | 000B5 | PUSHL | #4522104 | | |
| | | 63 | | 02 | FB | 000BB | CALLS | #2, LIB\$STOP | | |
| | | 50 | | 01 | D0 | 000BE | MOVL | #1, R0 | 1285 | |
| | | | | 04 | 000C1 | | RET | | | |

; Routine Size: 194 bytes, Routine Base: \$CODE\$ + 0491

```
1300 1286 1 GLOBAL ROUTINE ACT_REMOVE =
1301 1287 1
1302 1288 1 ++
1303 1289 1
1304 1290 1 Functional Description:
1305 1291 1
1306 1292 1 This action routine implements the REMOVE command. It removes the
1307 1293 1 specified entry from the quota file.
1308 1294 1
1309 1295 1 Calling Sequence:
1310 1296 1 standard
1311 1297 1
1312 1298 1 Input Parameters:
1313 1299 1 none
1314 1300 1
1315 1301 1 Implicit Inputs:
1316 1302 1 none
1317 1303 1
1318 1304 1 Output Parameters:
1319 1305 1 none
1320 1306 1
1321 1307 1 Implicit Outputs:
1322 1308 1 none
1323 1309 1
1324 1310 1 Routines Called:
1325 1311 1 none
1326 1312 1
1327 1313 1 Routine Value:
1328 1314 1 none
1329 1315 1
1330 1316 1 Signals:
1331 1317 1 none
1332 1318 1
1333 1319 1 Side Effects:
1334 1320 1 none
1335 1321 1
1336 1322 1 --
1337 1323 1
1338 1324 2 BEGIN
1339 1325 2
1340 1326 2
1341 1327 2 LOCAL
1342 1328 2 STATUS; ! general status value
1343 1329 2
1344 1330 2
1345 1331 2 ! Verify that a channel is open.
1346 1332 2
1347 1333 2
1348 1334 2 IF .CHANNEL EQL 0
1349 1335 2 THEN ERR_EXIT (DSKQS_NODEVICE);
1350 1336 2
1351 1337 2 ! Set any appropriate wildcard indicators.
1352 1338 2
1353 1339 2
1354 1340 2 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP THEN UIC_FLAGS[WILD_GROUP] = 1;
1355 1341 2 IF .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER THEN UIC_FLAGS[WILD_MEMBER] = 1;
1356 1342 2
```

```
1357 1343 2 ! Loop for all matching entries in the quota file, making a call to
1358 1344 2 ! remove each.
1359 1345 2 !
1360 1346 2 !
1361 1347 2 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_REM_QUOTA;
1362 1348 2 QUOTA_FIB[FIBSL_CNTRLVAL] = .UIC_FLAGS;
1363 1349 2 QUOTA_FIB[FIBSL_WCC] = 0;
1364 1350 2
1365 1351 2 INCR J FROM 0
1366 1352 2 DO
1367 1353 2 BEGIN
1368 1354 2
1369 1355 2 STATUS = $QIOW (CHAN = .CHANNEL,
1370 1356 2 FUNC = IOS_ACPCONTROL,
1371 1357 2 IOSB = IO_STATUS,
1372 1358 2 P1 = QFIB_DESC,
1373 1359 2 P2 = SRCREC_DESC,
1374 1360 2 P4 = DSTREC_DESC,
1375 1361 2 );
1376 1362 2 IF .STATUS THEN STATUS = .IO_STATUS[0];
1377 1363 2 IF .STATUS
1378 1364 2 THEN
1379 1365 2 BEGIN
1380 1366 2 IF .STATUS EQL SSS_OVRDSKQUOTA
1381 1367 2 THEN ERR_MESSAGE (DSKQ$_INUSE,
1382 1368 2 .(DST_REC[DQFSL_UIC])<16,16>,
1383 1369 2 .(DST_REC[DQFSL_UIC])<00,16>,
1384 1370 2 .DST_REC[DQFSL_USAGE]);
1385 1371 2 END
1386 1372 2 ELSE
1387 1373 2 BEGIN
1388 1374 2 IF .STATUS EQL SSS_NODISKQUOTA
1389 1375 2 AND .J NEQ 0
1390 1376 2 THEN EXITLOOP;
1391 1377 2 ERR_EXIT (DSKQ$_REMOVERR, .STATUS);
1392 1378 2 END;
1393 1379 2
1394 1380 2 IF NOT .UIC_FLAGS[WILD_GROUP]
1395 1381 2 AND NOT .UIC_FLAGS[WILD_MEMBER]
1396 1382 2 THEN EXITLOOP; ! done if no wild cards
1397 1383 2
1398 1384 2 END; ! end of loop
1399 1385 2
1400 1386 2 !
1401 1387 2 ! end of routine ACT_REMOVE
1387 1 END;
```

```
003C 00000
55 00000000G 00 9E 00002
54 00000000' EF 9E 00009
FE44 C4 B5 00010
09 12 00014
00450048 BF DD 00016
65 01 FB 0001C
```

```
.ENTRY ACT_REMOVE, Save R2,R3,R4,R5
MOVAB LIB$STOP, R5
MOVAB UIC_FLAGS, R4
TSTW CHANNEL
BNEQ 1$
PUSHL #4522056
CALLS #1, LIB$STOP
```

```
: 1286
:
: 1334
: 1335
:
```


| | | | | | | | | | |
|-----------|----|----------|----|-------|-------|-----|--------|-------------------------|------|
| 3FFF | 8F | 86 | A4 | B1 | 0001F | 18: | CMPL | UIC_VALUE+2, #16383 | 1340 |
| | | | 03 | 12 | 00025 | | BNEQ | 28 | |
| | 64 | | 02 | 88 | 00027 | | BISB2 | #2, UIC_FLAGS | |
| FFFF | 8F | 84 | A4 | B1 | 0002A | 28: | CMPL | UIC_VALUE, #65535 | 1341 |
| | | | 03 | 12 | 00030 | | BNEQ | 38 | |
| | 64 | | 01 | 88 | 00032 | | BISB2 | #1, UIC_FLAGS | |
| D6 | A4 | | 0E | B0 | 00035 | 38: | MOVW | #14, QUOTA_FIB+22 | 1347 |
| D8 | A4 | | 64 | D0 | 00039 | | MOVL | UIC_FLAGS, QUOTA_FIB+24 | 1348 |
| | | D0 | A4 | D4 | 0003D | | CLRL | QUOTA_FIB+16 | 1349 |
| | | | 53 | D4 | 00040 | | CLRL | J | 1351 |
| | | | 7E | 7C | 00042 | 48: | CLRL | -(SP) | 1361 |
| | | 48 | A4 | 9F | 00044 | | PUSHAB | DSTREC_DESC | |
| | | | 7E | D4 | 00047 | | CLRL | -(SP) | |
| | | 40 | A4 | 9F | 00049 | | PUSHAB | SRCREC_DESC | |
| | | 50 | A4 | 9F | 0004C | | PUSHAB | QFIB_DESC | |
| | | | 7E | 7C | 0004F | | CLRL | -(SP) | |
| | | FE48 | C4 | 9F | 00051 | | PUSHAB | IO_STATUS | |
| | | | 38 | DD | 00055 | | PUSHL | #58 | |
| | 7E | FE44 | C4 | 3C | 00057 | | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E | D4 | 0005C | | CLRL | -(SP) | |
| 00000000G | 00 | | 0C | FB | 0005E | | CALLS | #12, SYS\$QIOW | |
| | 52 | | 50 | D0 | 00065 | | MOVL | R0, STATUS | |
| | 2B | | 52 | E9 | 00068 | | BLBC | STATUS, 58 | 1362 |
| | 52 | FE48 | C4 | 3C | 0006B | | MOVZWL | IO_STATUS, STATUS | |
| | 23 | | 52 | E9 | 00070 | | BLBC | STATUS, 58 | 1363 |
| 00000669 | 8F | | 52 | D1 | 00073 | | CMPL | STATUS, #1641 | 1366 |
| | | | 32 | 12 | 0007A | | BNEQ | 78 | |
| | | A8 | A4 | DD | 0007C | | PUSHL | DST_REC+8 | 1370 |
| | 7E | A4 | A4 | 3C | 0007F | | MOVZWL | DST_REC+4, -(SP) | |
| | 7E | A6 | A4 | 3C | 00083 | | MOVZWL | DST_REC+6, -(SP) | |
| | | 00450098 | 8F | DD | 00087 | | PUSHL | #4522136 | |
| 00000000G | 00 | | 04 | FB | 0008D | | CALLS | #4, LIB\$SIGNAL | |
| | | | 18 | 11 | 00094 | | BRB | 78 | 1363 |
| 000003E4 | 8F | | 52 | D1 | 00096 | 58: | CMPL | STATUS, #996 | 1374 |
| | | | 04 | 12 | 0009D | | BNEQ | 68 | |
| | | | 53 | D5 | 0009F | | TSTL | J | 1375 |
| | | | 1A | 12 | 000A1 | | BNEQ | 98 | |
| | | 00450080 | 52 | DD | 000A3 | 68: | PUSHL | STATUS | 1377 |
| | 65 | | 8F | DD | 000A5 | | PUSHL | #4522112 | |
| 03 | 64 | | 02 | FB | 000AB | | CALLS | #2, LIB\$STOP | |
| | 08 | | 01 | E0 | 000AE | 78: | BBS | #1, UIC_FLAGS, 88 | 1380 |
| | | | 64 | E9 | 000B2 | | BLBC | UIC_FLAGS, 98 | 1381 |
| 85 | 53 | 7FFFFFFF | 8F | F3 | 000B5 | 88: | AOBLEQ | #2127483647, J, 48 | 1351 |
| | 50 | | 01 | D0 | 000BD | 98: | MOVL | #1, R0 | 1387 |
| | | | 04 | 000C0 | | | RET | | |

; Routine Size: 193 bytes, Routine Base: \$CODE\$ + 0553

```
1403 1388 1 GLOBAL ROUTINE ACT_SHOW =
1404 1389 1
1405 1390 1 **
1406 1391 1
1407 1392 1 Functional Description:
1408 1393 1
1409 1394 1     This action routine implements the SHOW command. It lists
1410 1395 1     UIC, quota, and usage of the indicated entries to SYS$OUTPUT.
1411 1396 1
1412 1397 1 Calling Sequence:
1413 1398 1     standard
1414 1399 1
1415 1400 1 Input Parameters:
1416 1401 1     none
1417 1402 1
1418 1403 1 Implicit Inputs:
1419 1404 1     none
1420 1405 1
1421 1406 1 Output Parameters:
1422 1407 1     none
1423 1408 1
1424 1409 1 Implicit Outputs:
1425 1410 1     none
1426 1411 1
1427 1412 1 Routines Called:
1428 1413 1     none
1429 1414 1
1430 1415 1 Routine Value:
1431 1416 1     none
1432 1417 1
1433 1418 1 Signals:
1434 1419 1     none
1435 1420 1
1436 1421 1 Side Effects:
1437 1422 1     none
1438 1423 1
1439 1424 1 --
1440 1425 1
1441 1426 2 BEGIN
1442 1427 2
1443 1428 2 BIND
1444 1429 2     LISTING_HEADER = DESCRIPTOR ('      UIC      Usage      Permanent Quota  Overdraft Limit'),
1445 1430 2     MULTI_FORMAT_1 = DESCRIPTOR ('!18<!AS!>!13<!UL!>!18<!UL!>!13<!UL!>'),
1446 1431 2     MULTI_FORMAT_2 = DESCRIPTOR ('!AS!/?!18* !13<!UL!>!18<!UL!>!13<!UL!>'),
1447 1432 2     SINGLE_FORMAT = DESCRIPTOR ('UIC !AS has !UL blocks used!/of !UL authorized, !UL permitted overdra
1448 1433 2
1449 1434 2
1450 1435 2 LOCAL
1451 1436 2     UIC_DESC : SBBLOCK [DESC S_BLN],          ! Descr for alpha UIC
1452 1437 2     FORMATTED_UIC : VECTOR [67,BYTE],          ! Alpha UIC storage
1453 1438 2     STATUS :          ! general status value
1454 1439 2
1455 1440 2 EXTERNAL ROUTINE
1456 1441 2     LIB$PUT_OUTPUT : ADDRESSING_MODE (GENERAL);
1457 1442 2
1458 1443 2
1459 1444 2 ! Verify that a channel is open.
```

```
1460 1445 !
1461 1446 !
1462 1447 IF .CHANNEL EQL 0
1463 1448 THEN ERR_EXIT (DSKQS_NODEVICE);
1464 1449 !
1465 1450 ! Set any appropriate wildcard indicators.
1466 1451 !
1467 1452 !
1468 1453 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP THEN UIC_FLAGS[WILD_GROUP] = 1;
1469 1454 IF .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER THEN UIC_FLAGS[WILD_MEMBER] = 1;
1470 1455 !
1471 1456 ! Loop for all matching entries in the quota file, making a call to
1472 1457 ! examine each.
1473 1458 !
1474 1459 !
1475 1460 QUOTA_FIB[FIB$W_CNTRLFUNC] = FIB$C_EXA_QUOTA;
1476 1461 QUOTA_FIB[FIB$S_CNTRLVAL] = .UIC_FLAGS;
1477 1462 QUOTA_FIB[FIB$S_WCC] = 0;
1478 1463 !
1479 1464 IF .UIC_FLAGS[WILD_GROUP]
1480 1465 OR .UIC_FLAGS[WILD_MEMBER]
1481 1466 THEN LIB$PUT_OUTPUT (LISTING_HEADER);
1482 1467 !
1483 1468 INCR J FROM 0
1484 1469 DO
1485 1470 BEGIN
1486 1471 STATUS = $QIOW (CHAN = .CHANNEL,
1487 1472 FUNC = IOS_ACPCONTROL,
1488 1473 IOSB = IO_STATUS,
1489 1474 P1 = QFIB_DESC,
1490 1475 P2 = SRCREC_DESC,
1491 1476 P4 = DSTREC_DESC
1492 1477 );
1493 1478 !
1494 1479 IF .STATUS THEN STATUS = .IO_STATUS[0];
1495 1480 IF NOT .STATUS
1496 1481 THEN
1497 1482 BEGIN
1498 1483 IF .STATUS EQL $$$_NODISKQUOTA
1499 1484 AND .J NEQ 0
1500 1485 THEN EXITLOOP;
1501 1486 ERR_EXIT (DSKQS_EXAMINERR, .STATUS);
1502 1487 END;
1503 1488 !
1504 1489 ! Format a listing line and output it.
1505 1490 !
1506 1491 !
1507 1492 UIC_DESC[DSC$W_LENGTH] = 67;
1508 1493 UIC_DESC[DSC$A_POINTER] = FORMATTED_UIC;
1509 1494 $FAD ($DESCRIPTOR ('!%I'),
1510 1495 UIC_DESC, UIC_DESC,
1511 1496 .DST_REC[DQF$C_UIC]);
1512 1497 !
1513 1498 OUTPUT_DESC[0] = OUTPUT_LENGTH;
1514 1499 $FAD (
1515 1500 (
1516 1501 IF .UIC_FLAGS[WILD_GROUP]
```

```
1517 OR .UIC_FLAGS[WILD_MEMBER]
1518 THEN (IF .UIC_DESC[DESC$W_LENGTH] LSS 18
1519 THEN MULTI_FORMAT_1
1520 ELSE MULTI_FORMAT_2)
1521 ELSE SINGLE_FORMAT),
1522 OUTPUT_DESC[0],
1523 OUTPUT_DESC[0],
1524 UIC_DESC,
1525 .DST_REC[DQF$USAGE],
1526 .DST_REC[DQF$PERMQUOTA],
1527 .DST_REC[DQF$OVERDRAFT]
1528 );
1529
1530 LIB$PUT_OUTPUT (OUTPUT_DESC[0]);
1531
1532 IF NOT .UIC_FLAGS[WILD_GROUP]
1533 AND NOT .UIC_FLAGS[WILD_MEMBER]
1534 THEN EXITLOOP; ! done if no wild cards
1535
1536 END; ! end of loop
1537
1538 1
1539 1 END; ! end of routine ACT_SHOW
```

```
20 20 20 20 20 20 20 43 49 55 20 20 20 20 20 0031C P.AAK: .ASCII \ UIC Usage Permanent\
20 20 20 20 20 20 20 65 67 61 73 55 20 20 20 0032B
72 64 72 65 76 4F 20 20 20 61 74 6F 75 51 20 0033A
74 6E 65 6E 61 6D 72 65 50 20 00344 .ASCII \ Quota Overdraft Limit\
74 69 6D 69 4C 20 74 66 61 00353
00000040 0035C P.AAJ: .LONG 64
00000000 00360 .ADDRESS P.AAK
55 21 3C 33 31 21 3E 21 53 41 21 3C 38 31 21 00364 P.AAM: .ASCII \!18<!AS!>!13<!UL!>!18<!UL!>!13<!UL!>\
33 31 21 3E 21 4C 55 21 3C 38 31 21 3E 21 4C 00373
3E 21 4C 55 21 3C 00382
00000024 00388 P.AAL: .LONG 36
00000000 0038C .ADDRESS P.AAM
21 3C 33 31 21 20 2A 38 31 21 2F 21 53 41 21 00390 P.AAO: .ASCII \!AS!/?!18* !13<!UL!>!18<!UL!>!13<!UL!>\
31 21 3E 21 4C 55 21 3C 38 31 21 3E 21 4C 55 0039F
3E 21 4C 55 21 3C 33 003AE
00000025 003B5 .BLKB 3
00000000 003B8 P.AAN: .LONG 37
00000000 003BC .ADDRESS P.AAO
4C 55 21 20 73 61 68 20 53 41 21 20 43 49 55 003C0 P.AAQ: .ASCII \UIC !AS has !UL blocks used!/of !UL auth\
6F 2F 21 64 65 73 75 20 73 68 63 6F 6C 62 20 003CF
68 74 75 61 20 4C 55 21 20 66 003DE
72 65 70 20 4C 55 21 20 2C 64 65 7A 69 72 6F 003E8
66 61 72 64 72 65 76 6F 20 64 65 74 74 69 6D 003F7
2E 74 00406
00000048 0040B P.AAP: .LONG 72
00000000 0040C .ADDRESS P.AAQ
49 25 21 00410 P.AAS: .ASCII \!X!\
00413 .BLKB 1
00000003 00414 P.AAR: .LONG 3
```


00000000' 00418

.ADDRESS P.AAS

LISTING HEADER= P.AAJ
MULTI-FORMAT-1= P.AAL
MULTI-FORMAT-2= P.AAN
SINGLE-FORMAT= P.AAP
.EXTRN LIB\$PUT_OUTPUT, SYS\$FAO

.PSECT \$CODE\$,NOWRT,2

| | | | | | | | |
|-----------|-----------|----------|------|----------|------------|-------------------------------------|------|
| | | | 01FC | 00000 | .ENTRY | ACT SHOW, Save R2,R3,R4,R5,R6,R7,R8 | 1388 |
| 58 | 00000000G | 00 | 9E | 00002 | MOVAB | SYS\$FAO, R8 | |
| 57 | 00000000G | 00 | 9E | 00009 | MOVAB | LIB\$PUT_OUTPUT, R7 | |
| 56 | 00000000G | 00 | 9E | 00010 | MOVAB | LIB\$STOP, R6 | |
| 55 | 00000000' | EF | 9E | 00017 | MOVAB | LISTING HEADER, R5 | |
| 54 | 00000000' | EF | 9E | 0001E | MOVAB | UIC_FLAGS, R4 | |
| 5E | B4 | AE | 9E | 00025 | MOVAB | -76(TSP), SP | |
| | FE44 | C4 | 85 | 00029 | TSTW | CHANNEL | 1447 |
| | | 09 | 12 | 0002D | BNEQ | 1\$ | |
| | 00450048 | 8F | DD | 0002F | PUSHL | #4522056 | 1448 |
| | | 01 | FB | 00035 | CALLS | #1, LIB\$STOP | |
| 3FFF | 66 | B6 | A4 | B1 00038 | 1\$: CMPW | UIC_VALUE+2, #16383 | 1453 |
| | 8F | | 03 | 12 0003E | BNEQ | 2\$ | |
| | 64 | | 02 | 88 00040 | BISB2 | #2, UIC_FLAGS | |
| FFFF | 8F | B4 | A4 | B1 00043 | 2\$: CMPW | UIC_VALUE, #65535 | 1454 |
| | | | 03 | 12 00049 | BNEQ | 3\$ | |
| | 64 | | 01 | 88 0004B | BISB2 | #1, UIC_FLAGS | |
| D6 | A4 | | 0C | B0 0004E | 3\$: MOVW | #12, QUOTA_FIB+22 | 1460 |
| DB | A4 | | 64 | D0 00052 | MOVL | UIC_FLAGS, QUOTA_FIB+24 | 1461 |
| | | DD | A4 | D4 00056 | CLRL | QUOTA_FIB+16 | 1462 |
| 03 | | | 01 | E0 00059 | BBS | #1, UIC_FLAGS, 4\$ | 1464 |
| | 64 | | 64 | E9 0005D | BLBC | UIC_FLAGS, 5\$ | 1465 |
| | 05 | | 55 | DD 00060 | 4\$: PUSHL | R5 | 1466 |
| | | | 01 | FB 00062 | CALLS | #1, LIB\$PUT_OUTPUT | |
| | 67 | | 53 | D4 00065 | 5\$: CLRL | J | 1468 |
| | | | 7E | 7C 00067 | 6\$: CLRQ | -(SP) | 1478 |
| | | 4B | A4 | 9F 00069 | PUSHAB | DSTREC_DESC | |
| | | | 7E | D4 0006C | CLRL | -(SP) | |
| | | 40 | A4 | 9F 0006E | PUSHAB | SRCREC_DESC | |
| | | 50 | A4 | 9F 00071 | PUSHAB | QFIB_DESC | |
| | | | 7E | 7C 00074 | CLRQ | -(SP) | |
| | | FE48 | C4 | 9F 00076 | PUSHAB | IO STATUS | |
| | | | 3B | DD 0007A | PUSHL | #5\$ | |
| | 7E | FE44 | C4 | 3C 0007C | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E | D4 00081 | CLRL | -(SP) | |
| 00000000G | 00 | | 0C | FB 00083 | CALLS | #12, SYS\$Q10W | |
| | 52 | | 50 | D0 0008A | MOVL | R0, STATUS | |
| | 08 | | 52 | E9 0008D | BLBC | STATUS, 7\$ | 1479 |
| | 52 | FE48 | C4 | 3C 00090 | MOVZWL | IO STATUS, STATUS | |
| | 18 | | 52 | E8 00095 | BLBS | STATUS, 9\$ | 1480 |
| 000003E4 | 8F | | 52 | D1 00098 | 7\$: CMPL | STATUS, #996 | 1483 |
| | | | 04 | 12 0009F | BNEQ | 8\$ | |
| | | | 53 | D5 000A1 | TSTL | J | 1484 |
| | | | 77 | 12 000A3 | BNEQ | 15\$ | |
| | | | 52 | DD 000A5 | 8\$: PUSHL | STATUS | 1486 |
| | | 00450090 | 8F | DD 000A7 | PUSHL | #4522128 | |
| 66 | | | 02 | FB 000AD | CALLS | #2, LIB\$STOP | |

| | | | | | | | | | | |
|------|----|------|----------|----|-------|-------|--------|---------------------------|-------------------------|------|
| 44 | AE | 43 | 8F | 9B | 000B0 | 9%: | MOVZBW | #67, UIC_DESC | 1492 | |
| 48 | AE | | 6E | 9E | 000B5 | | MOVAB | FORMATTED_UIC, UIC_DESC+4 | 1493 | |
| | | A4 | A4 | DD | 000B9 | | PUSHL | DST_REC+4 | 1496 | |
| | | 48 | AE | 9F | 000BC | | PUSHAB | UIC_DESC | | |
| | | 4C | AE | 9F | 000BF | | PUSHAB | UIC_DESC | | |
| | | 00B8 | C5 | 9F | 000C2 | | PUSHAB | P.AAR | | |
| | 68 | | 04 | FB | 000C6 | | CALLS | #4, SYSS\$FAO | | |
| FF60 | C4 | 84 | 8F | 9A | 000C9 | | MOVZBL | #132, OUTPUT_DESC | 1498 | |
| | 7E | AC | A4 | 7D | 000CF | | MOVQ | DST_REC+12, =(SP) | 1513 | |
| | | A8 | A4 | DD | 000D3 | | PUSHL | DST_REC+8 | | |
| | | 50 | AE | 9F | 000D6 | | PUSHAB | UIC_DESC | | |
| | | FF60 | C4 | 9F | 000D9 | | PUSHAB | OUTPUT_DESC | | |
| | | FF60 | C4 | 9F | 000DD | | PUSHAB | OUTPUT_DESC | | |
| 03 | 64 | | 01 | E0 | 000E1 | | BBS | #1, UIC_FLAGS, 10\$ | | |
| | 12 | | 64 | E9 | 000E5 | | BLBC | UIC_FLAGS, 12\$ | | |
| | 12 | 5C | AE | B1 | 000E8 | 10%: | CMPW | UIC_DESC, #18 | | |
| | | | 06 | 1E | 000EC | | BGEQU | 11\$ | | |
| | 50 | 2C | A5 | 9E | 000EE | | MOVAB | MULTI_FORMAT_1, R0 | | |
| | | | 0B | 11 | 000F2 | | BRB | 13\$ | | |
| | 50 | 5C | A5 | 9E | 000F4 | 11%: | MOVAB | MULTI_FORMAT_2, R0 | | |
| | | | 05 | 11 | 000F8 | | BRB | 13\$ | | |
| | 50 | 00AC | C5 | 9E | 000FA | 12%: | MOVAB | SINGLE_FORMAT, R0 | | |
| | | | 50 | DD | 000FF | 13%: | PUSHL | R0 | | |
| | 68 | | 07 | FB | 00101 | | CALLS | #7, SYSS\$FAO | | |
| | | FF60 | C4 | 9F | 00104 | | PUSHAB | OUTPUT_DESC | 1515 | |
| | 67 | | 01 | FB | 00108 | | CALLS | #1, LIB\$PUT_OUTPUT | | |
| 03 | 64 | | 01 | E0 | 0010B | | BBS | #1, UIC_FLAGS, 14\$ | 1517 | |
| | 0A | | 64 | E9 | 0010F | | BLBC | UIC_FLAGS, 15\$ | 1518 | |
| FF4B | 53 | 01 | 7FFFFFFF | 8F | F1 | 00112 | 14%: | ACBL | #2147483647, #1, J, 6\$ | 1468 |
| | | 50 | | 01 | DD | 0011C | 15%: | MOVL | #1, R0 | 1524 |
| | | | | 04 | 0011F | | RET | | | |

; Routine Size: 288 bytes, Routine Base: \$CODE\$ + 0614

```
1541 1525 1 GLOBAL ROUTINE ACT_MODIFY =
1542 1526 1
1543 1527 1 ++
1544 1528 1
1545 1529 1 Functional Description:
1546 1530 1
1547 1531 1 This action routine implements the MODIFY command. It modifies the
1548 1532 1 specified entry of the quota file as specified.
1549 1533 1
1550 1534 1 Calling Sequence:
1551 1535 1 standard
1552 1536 1
1553 1537 1 Input Parameters:
1554 1538 1 none
1555 1539 1
1556 1540 1 Implicit Inputs:
1557 1541 1 none
1558 1542 1
1559 1543 1 Output Parameters:
1560 1544 1 none
1561 1545 1
1562 1546 1 Implicit Outputs:
1563 1547 1 none
1564 1548 1
1565 1549 1 Routines Called:
1566 1550 1 none
1567 1551 1
1568 1552 1 Routine Value:
1569 1553 1 none
1570 1554 1
1571 1555 1 Signals:
1572 1556 1 none
1573 1557 1
1574 1558 1 Side Effects:
1575 1559 1 none
1576 1560 1
1577 1561 1 --
1578 1562 1
1579 1563 2 BEGIN
1580 1564 2
1581 1565 2
1582 1566 2 LOCAL
1583 1567 2 STATUS; ! general status value
1584 1568 2
1585 1569 2
1586 1570 2 ! Verify that a channel is open.
1587 1571 2
1588 1572 2
1589 1573 2 IF .CHANNEL EQL 0
1590 1574 2 THEN ERR_EXIT (DSK0$_NODEVICE);
1591 1575 2
1592 1576 2 ! Set any appropriate wildcard indicators.
1593 1577 2
1594 1578 2
1595 1579 2 IF .UIC_VALUE<16,16> EQL UIC$K_WILD_GROUP THEN UIC_FLAGS[WILD_GROUP] = 1;
1596 1580 2 IF .UIC_VALUE<0,16> EQL UIC$K_WILD_MEMBER THEN UIC_FLAGS[WILD_MEMBER] = 1;
1597 1581 2
```

```
1598      1582      2      ! Loop for all matching entries in the quota file, making a call to
1599      1583      2      ! modify each.
1600      1584      2      !
1601      1585      2      !
1602      1586      2      QUOTA_FIB[FIBSW_CNTRLFUNC] = FIBSC_MCD_QUOTA;
1603      1587      2      QUOTA_FIB[FIBSL_CNTRLVAL] = .UIC_FLAGS;
1604      1588      2      QUOTA_FIB[FIBSL_WCC] = 0;
1605      1589      2      !
1606      1590      2      INCR J FROM 0
1607      1591      2      DO
1608      1592      2      BEGIN
1609      1593      2      STATUS = $QIOW (CHAN = .CHANNEL,
1610      1594      2      FUNC = IOS_ACPCONTROL,
1611      1595      2      IOSB = IO_STATUS,
1612      1596      2      P1 = QFIB_DESC,
1613      1597      2      P2 = SRCREC_DESC,
1614      1598      2      P4 = DSTREC_DESC
1615      1599      2      );
1616      1600      2      IF .STATUS THEN STATUS = .IO_STATUS[0];
1617      1601      2      IF .STATUS
1618      1602      2      THEN
1619      1603      2      BEGIN
1620      1604      2      IF .STATUS EQL SSS_OVRDSKQUOTA
1621      1605      2      THEN ERR_MESSAGE (DSKQ$_INUSE,
1622      1606      2      .(DST_REC[DQFSL_UIC])<16,16>,
1623      1607      2      .(DST_REC[DQFSL_UIC])<00,16>,
1624      1608      2      .DST_REC[DQFSL_USAGE]);
1625      1609      2      END
1626      1610      2      ELSE
1627      1611      2      BEGIN
1628      1612      2      IF .STATUS EQL SSS_NODISKQUOTA
1629      1613      2      AND .J NEQ 0
1630      1614      2      THEN EXITLOOP;
1631      1615      2      ERR_EXIT (DSKQ$_MODIFYERR, .STATUS);
1632      1616      2      END;
1633      1617      2      !
1634      1618      2      !
1635      1619      2      IF NOT .UIC_FLAGS[WILD_GROUP]
1636      1620      2      AND NOT .UIC_FLAGS[WILD_MEMBER]
1637      1621      2      THEN EXITLOOP;
1638      1622      2      ! done if no wild cards
1639      1623      2      !
1640      1624      2      ! end of loop
1641      1625      2      !
1642      1626      2      ! end of routine ACT_MODIFY
```

```
003C 00000
55 00000000G 00 9E 00002
54 00000C00' EF 9E 00009
FE44 C4 B5 00010
09 12 00014
00450048 8F DD 00016
65 01 FB 0001C
```

```
.ENTRY ACT_MODIFY, Save R2,R3,R4,R5
MOVAB LIB$STOP, R5
MOVAB UIC_FLAGS, R4
TSTW CHANNEL
BNEQ 18
PUSHL #4522056
CALLS #1, LIB$STOP
```

```
: 1525
:
: 1573
: 1574
:
```


| | | | | | | | | | |
|-----------|----|----------|----|----|-------|------|--------|-------------------------|------|
| 3FFF | 8F | 86 | A4 | B1 | 0001F | 1\$: | CMPL | UIC_VALUE+2, #16383 | 1579 |
| | | | 03 | 12 | 00025 | | BNEQ | 2\$ | |
| | 64 | | 02 | 88 | 00027 | | BISB2 | #2, UIC_FLAGS | |
| FFFF | 8F | 84 | A4 | B1 | 0002A | 2\$: | CMPL | UIC_VALUE, #65535 | 1580 |
| | | | 03 | 12 | 00030 | | BNEQ | 3\$ | |
| | 64 | | 01 | 88 | 00032 | | BISB2 | #1, UIC_FLAGS | |
| D6 | A4 | | 0D | 80 | 00035 | 3\$: | MOVW | #13, QUOTA_FIB+22 | 1586 |
| D8 | A4 | | 64 | D0 | 00039 | | MOVL | UIC_FLAGS, QUOTA_FIB+24 | 1587 |
| | | D0 | A4 | D4 | 0003D | | CLRL | QUOTA_FIB+16 | 1588 |
| | | | 53 | D4 | 00040 | | CLRL | J | 1590 |
| | | | 7E | 7C | 00042 | 4\$: | CLRL | -(SP) | 1600 |
| | | 48 | A4 | 9F | 00044 | | PUSHAB | DSTREC_DESC | |
| | | | 7E | D4 | 00047 | | CLRL | -(SP) | |
| | | 40 | A4 | 9F | 00049 | | PUSHAB | SRCREC_DESC | |
| | | 50 | A4 | 9F | 0004C | | PUSHAB | QFIB_DESC | |
| | | | 7E | 7C | 0004F | | CLRL | -(SP) | |
| | | FE48 | C4 | 9F | 00051 | | PUSHAB | IO_STATUS | |
| | | | 38 | DD | 00055 | | PUSHL | #58 | |
| | 7E | FE44 | C4 | 3C | 00057 | | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E | D4 | 0005C | | CLRL | -(SP) | |
| 00000000G | 00 | | 0C | FB | 0005E | | CALLS | #12, SYS\$QIOW | |
| | 52 | | 50 | D0 | 00065 | | MOVL | R0, STATUS | |
| | 2B | | 52 | E9 | 00068 | | BLBC | STATUS, 5\$ | 1601 |
| | 52 | FE48 | C4 | 3C | 0006B | | MOVZWL | IO_STATUS, STATUS | |
| | 23 | | 52 | E9 | 00070 | | BLBC | STATUS, 5\$ | 1602 |
| 00000669 | 8F | | 52 | D1 | 00073 | | CMPL | STATUS, #1641 | 1605 |
| | | | 32 | 12 | 0007A | | BNEQ | 7\$ | |
| | | A8 | A4 | DD | 0007C | | PUSHL | DST_REC+8 | 1609 |
| | 7E | | A4 | 3C | 0007F | | MOVZWL | DST_REC+4, -(SP) | |
| | 7E | | A4 | 3C | 00083 | | MOVZWL | DST_REC+6, -(SP) | |
| | | 00450098 | 8F | DD | 00087 | | PUSHL | #4522136 | |
| 00000000G | 00 | | 04 | FB | 0008D | | CALLS | #4, LIB\$SIGNAL | |
| | | | 18 | 11 | 00094 | | BRB | 7\$ | 1602 |
| 000003E4 | 8F | | 52 | D1 | 00096 | 5\$: | CMPL | STATUS, #996 | 1613 |
| | | | 04 | 12 | 0009D | | BNEQ | 6\$ | |
| | | | 53 | D5 | 0009F | | TSTL | J | 1614 |
| | | | 1A | 12 | 000A1 | | BNEQ | 9\$ | |
| | | 00450088 | 52 | DD | 000A3 | 6\$: | PUSHL | STATUS | 1616 |
| | 65 | | 8F | DD | 000A5 | | PUSHL | #4522120 | |
| 03 | 64 | | 02 | FB | 000AB | | CALLS | #2, LIB\$STOP | |
| | 08 | | 01 | E0 | 000AE | 7\$: | BBS | #1, UIC_FLAGS, 8\$ | 1619 |
| | | | 64 | E9 | 000B2 | | BLBC | UIC_FLAGS, 9\$ | 1620 |
| 85 | 53 | 7FFFFFFF | 8F | F3 | 000B5 | 8\$: | AOBLEQ | #2147483647, J, 4\$ | 1590 |
| | 50 | | 01 | D0 | 000BD | 9\$: | MOVL | #1, R0 | 1626 |
| | | | 04 | 00 | 000C0 | | RET | | |

; Routine Size: 193 bytes, Routine Base: \$CODE\$ + 0734

```
1644 1627 1 GLOBAL ROUTINE ACT_REBUILD =
1645 1628 1
1646 1629 1 **
1647 1630 1
1648 1631 1 Functional Description:
1649 1632 1
1650 1633 1 This routine implements the REBUILD command. It scans the index file
1651 1634 1 of each volume in the volume set and constructs a table of UIC's
1652 1635 1 and blocks used. It then updates the usage data in the quota file,
1653 1636 1 creating entries as needed so that all UIC's using blocks are listed.
1654 1637 1
1655 1638 1 Calling Sequence:
1656 1639 1 standard
1657 1640 1
1658 1641 1 Input Parameters:
1659 1642 1 none
1660 1643 1
1661 1644 1 Implicit Inputs:
1662 1645 1 none
1663 1646 1
1664 1647 1 Output Parameters:
1665 1648 1 none
1666 1649 1
1667 1650 1 Implicit Outputs:
1668 1651 1 none
1669 1652 1
1670 1653 1 Routines Called:
1671 1654 1 none
1672 1655 1
1673 1656 1 Routine Value:
1674 1657 1 none
1675 1658 1
1676 1659 1 Signals:
1677 1660 1 none
1678 1661 1
1679 1662 1 Side Effects:
1680 1663 1 none
1681 1664 1
1682 1665 1 --
1683 1666 1
1684 1667 2 BEGIN
1685 1668 2
1686 1669 2 LOCAL
1687 1670 2 STATUS; ! general status value
1688 1671 2
1689 1672 2 EXTERNAL ROUTINE
1690 1673 2 REBUILD : ADDRESSING_MODE (GENERAL); ! routine to do actual rebuild
1691 1674 2
1692 1675 2
1693 1676 2 ! Verify that a channel is open.
1694 1677 2
1695 1678 2
1696 1679 2 IF .CHANNEL EQL 0
1697 1680 2 THEN ERR_EXIT (DSKQ$_NODEVICE);
1698 1681 2
1699 1682 2 ! Enable the quota file, just in case it is off.
1700 1683 2
```

```
1701 1684 2
1702 1685 QUOTA_FIB[FIBSW_DID_NUM] = FIDSC_MFD;
1703 1686 QUOTA_FIB[FIBSW_DID_SEQ] = FIDSC_MFD;
1704 1687 QUOTA_FIB[FIBSW_DID_RVN] = 1;
1705 1688 QUOTA_FIB[FIBSW_CNTRLFUNC] = FIDSC_ENA_QUOTA;
1706 1689 STATUS = $QIOW (CHAN = .CHANNEL,
1707 1690 FUNC = IOS_ACPCONTROL,
1708 1691 IOSB = IO_STATUS,
1709 1692 P1 = QFIB_DESC,
1710 1693 P2 = QFILE_NAME
1711 1694 );
1712 1695 IF .STATUS THEN STATUS = .IO_STATUS[0];
1713 1696 IF NOT .STATUS
1714 1697 AND .STATUS NEQ $$$QFACTIVE
1715 1698 THEN ERR_EXIT (DSKQ$ACTERR, .STATUS);
1716 1699
1717 1700 ! Now call the rebuild routine.
1718 1701 !
1719 1702
1720 1703 REBUILD (.CHANNEL, 1);
1721 1704
1722 1705 1
1723 1706 1 END;

! end of routine ACT_REBUILD
```

| | | | | .EXTRN | REBUILD | |
|-----------|------|-----------|-------------|--------|-------------------------|------|
| | | | 000C 00000 | .ENTRY | ACT REBUILD, Save R2,R3 | 1627 |
| | 53 | 00000000G | 00 9E 00002 | MOVAB | LIB\$STOP, R3 | |
| | 52 | 00000000' | EF 9E 00009 | MOVAB | CHANNEL, R2 | |
| | | | 62 B5 00010 | TSTW | CHANNEL | 1679 |
| | | | 09 12 00012 | BNEQ | 1\$ | |
| | | 00450048 | 8F DD 00014 | PUSHL | #4522056 | 1680 |
| | 63 | | 01 FB 0001A | CALLS | #1, LIB\$STOP | |
| 0186 | C2 | 00040004 | 8F D0 0001D | MOVL | #262148, QUOTA_FIB+10 | 1685 |
| 018A | C2 | | 01 B0 00026 | MOVW | #1, QUOTA_FIB+T4 | 1687 |
| 0192 | C2 | | 09 B0 0002B | MOVW | #9, QUOTA_FIB+22 | 1688 |
| | | | 7E 7C 00030 | CLRQ | -(SP) | 1694 |
| | | | 7E 7C 00032 | CLRQ | -(SP) | |
| | 0244 | | C2 9F 00034 | PUSHAB | QFILE_NAME | |
| | 020C | | C2 9F 00038 | PUSHAB | QFIB_DESC | |
| | | | 7E 7C 0003C | CLRQ | -(SP) | |
| | 04 | | A2 9F 0003E | PUSHAB | IO_STATUS | |
| | | | 38 DD 00041 | PUSHL | #56 | |
| | 7E | | 62 3C 00043 | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E D4 00046 | C'RL | -(SP) | |
| 00000000G | 00 | | 0C FB 00048 | CALLS | #12, SYSSQIOW | |
| | 07 | | 50 E9 0004F | BLBC | STATUS, 2\$ | 1695 |
| | 50 | 04 | A2 3C 00052 | MOVZWL | IO_STATUS, STATUS | |
| | 14 | | 50 E8 00056 | BLBS | STATUS, 3\$ | 1696 |
| 000003CC | 8F | | 50 D1 00059 | CMPL | STATUS, #972 | 1697 |
| | | | 0B 13 00060 | BEQL | 3\$ | |
| | | | 50 DD 00062 | PUSHL | STATUS | 1698 |
| | | 00450068 | 8F DD 00064 | PUSHL | #4522088 | |
| 63 | | | 02 FB 0006A | CALLS | #2, LIB\$STOP | |
| | | | 01 DD 0006D | PUSHL | #1 | 1703 |

K 14
15-Sep-1984 23:38:38 YAX-11 B11ss-32 V4.0-742
14-Sep-1984 12:19:46 [DISKQ.SRC]DISKQUOTA.B32;1

Page 68
(18)

| | | | | | | |
|-----------|----|----|----|-------|--------|----------------|
| 00000000G | 7E | 62 | 3C | 0006F | MOVZWL | CHANNEL, -(SP) |
| | 00 | 02 | FB | 00072 | CALLS | #2, REBUILD |
| | 50 | 01 | D0 | 00079 | MOVL | #1, R0 |
| | | | 04 | 0007C | RET | |

1706

; Routine Size: 125 bytes, Routine Base: \$CODES + 07F5


```
1725 1707 1 GLOBAL ROUTINE ACT_HELP =
1726 1708 1 ++
1727 1709 1
1728 1710 1 Functional Description:
1729 1711 1
1730 1712 1 This routine is the DISKQUOTA help facility, and will display
1731 1713 1 useful and informative explanations of the DISKQUOTA facility.
1732 1714 1
1733 1715 1 To speed things up, the help library is opened only once, and
1734 1716 1 is closed by the OS during image rundown.
1735 1717 1
1736 1718 1 Calling Sequence:
1737 1719 1 standard
1738 1720 1
1739 1721 1 Input Parameters:
1740 1722 1 none
1741 1723 1
1742 1724 1 Implicit Inputs:
1743 1725 1
1744 1726 1 This routine expects the keys used to access the help text to
1745 1727 1 be in KEY_VECTOR[0..MAX_KEYS].
1746 1728 1
1747 1729 1 Output Parameters:
1748 1730 1 none
1749 1731 1
1750 1732 1 Implicit Outputs:
1751 1733 1
1752 1734 1 The help text will be printed on SYS$OUTPUT.
1753 1735 1
1754 1736 1 Routines Called:
1755 1737 1
1756 1738 1 LBR$INI CONTROL
1757 1739 1 LBR$OPEN
1758 1740 1 LBR$GET_HELP
1759 1741 1
1760 1742 1 Routine Value:
1761 1743 1 none
1762 1744 1
1763 1745 1 Signals:
1764 1746 1 none
1765 1747 1
1766 1748 1 Side Effects:
1767 1749 1 none
1768 1750 1
1769 1751 1 --
1770 1752 1
1771 1753 2 BEGIN
1772 1754 2
1773 1755 2 EXTERNAL ROUTINE
1774 1756 2
1775 1757 2 LBR$INI CONTROL : ADDRESSING_MODE(GENERAL),
1776 1758 2 LBR$OPEN : ADDRESSING_MODE(GENERAL),
1777 1759 2 LBR$GET_HELP : ADDRESSING_MODE(GENERAL);
1778 1760 2
1779 1761 2 BIND
1780 1762 2
1781 1763 2 HELP_DEFNAME = DESCRIPTOR ('SYS$HELP:.HLB'), ! default helpfile name
```

```
1782      LIBRARY_NAME = DESCRIPTOR ('DISKQUOTA');      ! HELP text library
1783
1784      OWN
1785
1786      HELP_FUNCTION : INITIAL (LBR$C_READ),
1787      HELP_TYPE     : INITIAL (LBR$C_TYP_HLP),      ! declare lib a HELP lib
1788      HELP_LIBINDEX : LONG,                        ! pointer to lib index
1789      LIBRARY_OPEN  : LONG;                        ! used as a boolean
1790
1791      LOCAL
1792
1793      STATUS;                                     ! used as boolean
1794
1795      !
1796      ! Check to see if HELPLIB is already OPENed. If it is, skip the
1797      ! OPENing code and get right to the HELP text retrieval.
1798
1799      IF NOT (.LIBRARY_OPEN)
1800      THEN
1801      BEGIN
1802      IF NOT (STATUS = LBR$INI_CONTROL (HELP_LIBINDEX, HELP_FUNCTION, HELP_TYPE))
1803      THEN
1804      ERR_EXIT (DSKQ$HELP_INIT, .STATUS);
1805
1806      IF NOT (STATUS = LBR$OPEN (HELP_LIBINDEX, LIBRARY_NAME, 0, HELP_DEFNAME))
1807      THEN
1808      ERR_EXIT (DSKQ$HELP_OPEN, .STATUS);
1809
1810      LIBRARY_OPEN = 1;                             ! flag library open
1811      END;
1812
1813      !
1814      ! Get and display the HELP text. LBR$GET_HELP will call LIB$PUT_OUTPUT
1815      ! to print the HELP text.
1816
1817      IF NOT (STATUS = LBR$GET_HELP (HELP_LIBINDEX, 0, 0, 0, KEY_VECTOR[0],
1818      KEY_VECTOR[2],
1819      KEY_VECTOR[4],
1820      KEY_VECTOR[6],
1821      KEY_VECTOR[8],
1822      KEY_VECTOR[10],
1823      KEY_VECTOR[12]))
1824      THEN
1825      ERR_EXIT (DSKQ$HELP_TEXT, .STATUS);
1826
1827      1
1828      !
1829      ! end of routine ACT_HELP
```

```
42 4C 48 2E 3A 50 4C 45 48 24 53 59 53 0041C P.AAU: .ASCII \SYS$HELP:.HLB\
0000000D 00429 .BLKB 3
00000000 0042C P.AAT: .LONG 13
00000000 00430 .ADDRESS P.AAU
```

```
.PSECT $PLITS,NOWRT,NOEXE,2
```

```
41 54 4F 55 51 4B 53 49 44 00434 P.AAW: .ASCII \DISKQUOTA\
                                0043D .BLKB 3
                                00000009 00440 P.AAV: .LONG 9
                                00000000 00444 .ADDRESS P.AAW
                                .PSECT SOWNS,NOEXE,2

00000001 0026C HELP_FUNCTION:
                                .LONG 1
00000003 00270 HELP_TYPE:
                                .LONG 3
                                00274 HELP_LIBINDEX:
                                .BLKB 4
                                00278 LIBRARY_OPEN:
                                .BLKB 4

HELP_DEFNAME= P.AAT
LIBRARY_NAME= P.AAV
                .EXTRN LBR$INI_CONTROL
                .EXTRN LBR$OPEN, LBR$GET_HELP
                .PSECT $CODE$,NOWRT,2

54 00000000G 001C 00000 .ENTRY ACT_HELP, Save R2,R3,R4
53 00000000' 00 9E 00002 MOVAB LBR$STOP, R4
4C 04 EF 9E 00009 MOVAB HELP_LIBINDEX, R3
FC A3 E8 00010 BLBS LIBRARY_OPEN, 3$
FB A3 9F 00014 PUSHAB HELP_TYPE
53 A3 9F 00017 PUSHAB HELP_FUNCTION
00 53 DD 0001A PUSHL R3
52 03 FB 0001C CALLS #3, LBR$INI_CONTROL
0B 50 D0 00023 MOVL R0, STATUS
04 52 E8 00026 BLBS STATUS, 1$
004500F0 52 DD 00029 PUSHL STATUS
64 8F DD 0002B PUSHL #4522224
00000000' 02 FB 00031 CALLS #2, LBR$STOP
7E EF 9F 00034 1$: PUSHAB HELP_DEFNAME
00000000' 7E D4 0003A CLRL -(SP)
00000000' EF 9F 0003C PUSHAB LIBRARY_NAME
00 53 DD 00042 PUSHL R3
52 04 FB 00044 CALLS #4, LBR$OPEN
0B 50 D0 0004B MOVL R0, STATUS
04 52 E8 0004E BLBS STATUS, 2$
004500F8 52 DD 00051 PUSHL STATUS
64 8F DD 00053 PUSHL #4522232
04 A3 02 FB 00059 CALLS #2, LBR$STOP
FF7C 01 D0 0005C 2$: MOVL #1, LIBRARY_OPEN
FF74 C3 9F 00060 3$: PUSHAB KEY_VECTOR+28
FF6C C3 9F 00064 PUSHAB KEY_VECTOR+40
FF64 C3 9F 00068 PUSHAB KEY_VECTOR+32
FF5C C3 9F 0006C PUSHAB KEY_VECTOR+24
FF54 C3 9F 00070 PUSHAB KEY_VECTOR+16
FF4C C3 9F 00074 PUSHAB KEY_VECTOR+8
C3 9F 00078 PUSHAB KEY_VECTOR
7E 7C 0007C CLRL -(SP)
7E D4 0007E CLRL -(SP)
53 DD 00080 PUSHL R3

1707
1782
1785
1787
1789
1791
1793
1806
1805
1804
1803
1802
1801
1800
```

```

0 15
15-Sep-1984 23:38:38      VAX-11 BLISS-32 V4.0-742
14-Sep-1984 12:19:46      [DISKQ.SRC]DISKQUOTA.B32;1

```

32

| Sym |
|-----|
| BUG |
| BUG |
| CLU |
| CTL |
| CTL |
| CTL |
| EXE |
| EXE |
| EXE |
| EXE |
| EXE |
| EXE |
| IN |
| IOC |
| IOC |
| IOC |
| KER |
| LIB |
| LNM |
| LNM |
| LOC |
| LOC |
| NSA |
| NSA |
| NSA |
| NSA |
| SCH |
| SCH |
| SCH |
| SCH |
| SCH |
| SEN |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| SYS |
| UNL |
| UNL |

; Routine Size: 158 bytes, Routine Base: \$CODES + 0872


```
1831 1812 1 GLOBAL ROUTINE MAIN_HANDLER (SIGNAL_VEC, MECHANISM) =
1832 1813 1
1833 1814 1 ++
1834 1815 1
1835 1816 1 Functional Description:
1836 1817 1
1837 1818 1 This routine is the main condition handler for the DISKQUOTA utility.
1838 1819 1 It receives a signal which is either an internal error code or a
1839 1820 1 standard system status. If the former, the appropriate message is
1840 1821 1 formatted and printed. For the latter, the condition is simply
1841 1822 1 resignalled.
1842 1823 1
1843 1824 1 Calling Sequence:
1844 1825 1 standard
1845 1826 1
1846 1827 1 Input Parameters:
1847 1828 1 none
1848 1829 1
1849 1830 1 Implicit Inputs:
1850 1831 1 none
1851 1832 1
1852 1833 1 Output Parameters:
1853 1834 1 none
1854 1835 1
1855 1836 1 Implicit Outputs:
1856 1837 1 none
1857 1838 1
1858 1839 1 Routines Called:
1859 1840 1 none
1860 1841 1
1861 1842 1 Routine Value:
1862 1843 1 none
1863 1844 1
1864 1845 1 Signals:
1865 1846 1 none
1866 1847 1
1867 1848 1 Side Effects:
1868 1849 1 none
1869 1850 1
1870 1851 1 --
1871 1852 1
1872 1853 2 BEGIN
1873 1854 2
1874 1855 2 MAP
1875 1856 2 SIGNAL_VEC : REF BBLOCK, ! signal vector arg
1876 1857 2 MECHANISM : REF BBLOCK; ! mechanism vector arg
1877 1858 2
1878 1859 2 LOCAL
1879 1860 2 FORMAT_DESC : VECTOR [2], ! string descriptor for message format
1880 1861 2 P : REF VECTOR [,BYTE], ! string pointer
1881 1862 2 ERR_CODE : BBLOCK [4]; ! error status code
1882 1863 2
1883 1864 2 EXTERNAL ROUTINE
1884 1865 2 LIB$PUT_OUTPUT : ADDRESSING_MODE (GENERAL);
1885 1866 2
1886 1867 2
1887 1868 2 ! Get the signal code. If it is one of ours, get the message string and
```

```
1888 1869 2 ! do formatting as necessary.
1889 1870 2
1890 1871 2
1891 1872 2 ERR_CODE = .SIGNAL_VEC[CHFSL_SIG_NAME];
1892 1873 2 IF .ERR_CODE[STSSV_FAC_NO] EQL FAC_CODE
1893 1874 2 THEN
1894 1875 2 BEGIN
1895 1876 2   ERR_CODE = .ERR_CODE[STSSV_MSG_NO];
1896 1877 2   P = .MESSAGE_TABLE[.ERR_CODE];
1897 1878 2   FORMAT_DESC[0] = .P[1];
1898 1879 2   FORMAT_DESC[1] = .P + 2;
1899 1880 2   OUTPUT_DESC[0] = OUTPUT_LENGTH;
1900 1881 2
1901 1882 2   $FAOL (CTRSTR = FORMAT_DESC[0],
1902 1883 2         OUTLEN = OUTPUT_DESC[0],
1903 1884 2         OUTBUF = OUTPUT_DESC[0],
1904 1885 2         PRMLST = SIGNAL_VEC[CHFSL_SIG_ARG1]
1905 1886 2         );
1906 1887 2   LIB$PUT_OUTPUT (OUTPUT_DESC);
1907 1888 2
1908 1889 2 ! If there is a signal argument remaining, it is a system error status.
1909 1890 2 ! Convert its severity to error and signal it.
1910 1891 2
1911 1892 2
1912 1893 2   ERR_CODE = 0;
1913 1894 2   IF .SIGNAL_VEC[CHFSL_SIG_ARGS] GTRU .P[0] + 3
1914 1895 2   THEN
1915 1896 2     BEGIN
1916 1897 2       ERR_CODE = .VECTOR [SIGNAL_VEC[CHFSL_SIG_ARG1], .P[0]];
1917 1898 2     END;
1918 1899 2   END;
1919 1900 2
1920 1901 2 IF .ERR_CODE NEQ 0
1921 1902 2 THEN
1922 1903 2 BEGIN
1923 1904 2   ERR_CODE[STSSV_SEVERITY] = STSSK_ERROR;
1924 1905 2   SIGNAL (.ERR_CODE);
1925 1906 2 END;
1926 1907 2
1927 1908 2 MECHANISM[CHFSL_MCH_SAVRO] = 1;
1928 1909 2 IF .BBLOCK [SIGNAL_VEC[CHFSL_SIG_NAME], STSSV_SEVERITY] EQL STSSK_SEVERE
1929 1910 2 THEN
1930 1911 2 BEGIN
1931 1912 2   $QIOW (CHAN = .CHANNEL,
1932 1913 2         FUNC = .IOB_DEACCESS);
1933 1914 2
1934 1915 2   $UNWIND (DEPADR = MECHANISM[CHFSL_MCH_DEPTH]);
1935 1916 2 END;
1936 1917 2
1937 1918 2 RETURN SSS_CONTINUE;
1938 1919 2
1939 1920 2 ! end of routine MAIN_HANDLER
```

.EXTRN SYS\$FAOL

| | | | | | | | | | | |
|----------|----|----|----|-----------|------|----|-------|--------|-----------------------------------|------|
| 00000045 | BF | 52 | 56 | 00000000' | EF | 9E | 00000 | .ENTRY | MAIN HANDLER, Save R2,R3,R4,R5,R6 | 1812 |
| | | | 5E | | 08 | C2 | 00009 | MOVAB | OUTPUT_DESC, R6 | |
| | | | 54 | 04 | AC | D0 | 0000C | SUBL2 | #8, SP | 1872 |
| | | | 52 | 04 | A4 | D0 | 00010 | MOVL | SIGNAL_VEC, R4 | |
| | | | 0C | | 10 | ED | 00014 | MOVL | 4(R4), ERR_CODE | 1873 |
| | | | | | 49 | 12 | 0001D | CMPZV | #16, #12, ERR_CODE, #69 | |
| | | | | | 03 | EF | 0001F | BNEQ | 1\$ | 1876 |
| | | | 0D | | 42 | D0 | 00024 | EXTZV | #3, #13, ERR_CODE, ERR_CODE | 1877 |
| | | | 53 | 00000000' | EF | 9A | 0002C | MOVL | MESSAGE_TABLE[ERR_CODE], P | 1878 |
| | | | 6E | 01 | A3 | 9A | 00030 | MOVZBL | 1(P), FORMAT_DESC | 1879 |
| | | | AE | 02 | A3 | 9E | 00035 | MOVAB | 2(R3), FORMAT_DESC+4 | 1880 |
| | | | 66 | 84 | BF | 9A | 00039 | MOVZBL | #132, OUTPUT_DESC | 1886 |
| | | | 55 | 08 | A4 | 9E | 0003D | MOVAB | 8(R4), R5 | |
| | | | | | 55 | DD | 0003F | PUSHL | R5 | |
| | | | | | 56 | DD | 00041 | PUSHL | R6 | |
| | | | | | 56 | DD | 00043 | PUSHL | R6 | |
| | | | | | AE | 9F | 00046 | PUSHAB | FORMAT_DESC | |
| | | | | | 04 | FB | 0004D | CALLS | #4, SYSSFAOL | 1887 |
| | | | | | 56 | DD | 0004F | PUSHL | R6 | |
| | | | | | 01 | FB | 00056 | CALLS | #1, LIB\$PUT_OUTPUT | 1893 |
| | | | | | 52 | D4 | 00058 | CLRL | ERR_CODE | 1894 |
| | | | | | 63 | 9A | 0005B | MOVZBL | (P), R0 | |
| | | | | | AO | 9E | 0005F | MOVAB | 3(R0), R1 | |
| | | | | | 64 | D1 | 00062 | CPL | (R4), R1 | |
| | | | | | 04 | 1B | 00064 | BLEQU | 1\$ | |
| | | | | | 6540 | D0 | 00068 | MOVL | (R5)[R0], ERR_CODE | 1897 |
| | | | | | 52 | D5 | 0006A | TSTL | ERR_CODE | 1901 |
| | | | | | 0E | 13 | 0006C | BEQL | 2\$ | |
| | | | | | 02 | F0 | 00071 | INSV | #2, #0, #3, ERR_CODE | 1904 |
| | | | | | 52 | DD | 00073 | PUSHL | ERR_CODE | 1905 |
| | | | | | 01 | FB | 0007A | CALLS | #1, LIB\$SIGNAL | |
| | | | | | AC | D0 | 0007E | MOVL | MECHANISM, R3 | 1908 |
| | | | | | 01 | D0 | 00082 | MOVL | #1, 12(R3) | |
| | | | | | 00 | ED | 00088 | CMPZV | #0, #3, 4(R4), #4 | 1909 |
| | | | | | 25 | 12 | 0008A | BNEQ | 3\$ | |
| | | | | | 7E | 7C | 0008C | CLRQ | -(SP) | 1913 |
| | | | | | 7E | 7C | 0008E | CLRQ | -(SP) | |
| | | | | | 7E | 7C | 00090 | CLRQ | -(SP) | |
| | | | | | 7E | 7C | 00092 | CLRQ | -(SP) | |
| | | | | | 34 | 7D | 00095 | MOVQ | #52, -(SP) | |
| | | | | | C6 | 3C | 0009A | MOVZWL | CHANNEL, -(SP) | |
| | | | | | 7E | D4 | 0009C | CLRL | -(SP) | |
| | | | | | 0C | FB | 000A3 | CALLS | #12, SYSSQIOW | 1915 |
| | | | | | 7E | D4 | 000A5 | CLRL | -(SP) | |
| | | | | | A3 | 9F | 000AB | PUSHAB | 8(R3) | |
| | | | | | 02 | FB | 000AF | CALLS | #2, SYSSUNWIND | 1918 |
| | | | | | 01 | D0 | 000B2 | MOVL | #1, R0 | 1920 |
| | | | | | 04 | 00 | 000B2 | RET | | |

; Routine Size: 179 bytes, Routine Base: \$CODE\$ + 0910

```
1941 1921 1 GLOBAL ROUTINE EXIT_HANDLER: NOVALUE =
1942 1922 1
1943 1923 1 **
1944 1924 1
1945 1925 1 Functional Description:
1946 1926 1
1947 1927 1 This routine is called by the OS on exit (for whatever reason) from
1948 1928 1 the DISKQUOTA utility. This routine must ensure that DISKQUOTA did
1949 1929 1 not leave things in an awkward state.
1950 1930 1
1951 1931 1 Calling Sequence:
1952 1932 1 standard
1953 1933 1
1954 1934 1 Input Parameters:
1955 1935 1 none
1956 1936 1
1957 1937 1 Implicit Inputs:
1958 1938 1 none
1959 1939 1
1960 1940 1 Output Parameters:
1961 1941 1 none
1962 1942 1
1963 1943 1 Implicit Outputs:
1964 1944 1 none
1965 1945 1
1966 1946 1 Routines Called:
1967 1947 1 none
1968 1948 1
1969 1949 1 Routine Value:
1970 1950 1 none
1971 1951 1
1972 1952 1 Signals:
1973 1953 1 none
1974 1954 1
1975 1955 1 Side Effects:
1976 1956 1 none
1977 1957 1
1978 1958 1 --
1979 1959 1
1980 1960 2 BEGIN
1981 1961 2
1982 1962 2
1983 1963 2 Make sure that DISKQUOTA did not leave a volume LOCKED.
1984 1964 2
1985 1965 2
1986 1966 2 IF .CLEANUP_FLAGS[CLF_UNLOCK]
1987 1967 2 THEN
1988 1968 2 BEGIN
1989 1969 2 CH$FILL (0, FIB$C_LENGTH, QUOTA_FIB);
1990 1970 2 QUOTA_FIB[FIB$W_CNTRLFUNC] = FIB$C_UNLK_VOL;
1991 1971 2 $QIOU(CHAN = .CHANNEL,
1992 1972 2 FUNC = IOS_ACPCONTROL,
1993 1973 2 P1 = QFIB_DESC
1994 1974 2 );
1995 1975 2 END;
1996 1976 2
1997 1977 1 END; ! end of routine EXIT_HANDLER
```


| | | | | | | | |
|------|----|----|--------------|---------------|--------|-----------------------------------|--------|
| 0040 | Bf | 00 | 56 00000000' | 007C 00000 | ENTRY | EXIT HANDLER, Save R2,R3,R4,R5,R6 | : 1921 |
| | | | 2A | EF 9E 00002 | MOVAB | CLEANUP_FLAGS, R6 | : 1966 |
| | | | 6E | 66 E9 00009 | BLBC | CLEANUP_FLAGS, 1\$ | : 1969 |
| | | | | 00 2C 0000C | MOVCS | #0, (SPT), #0, #64, QUOTA_FIB | : 1970 |
| | | | 44 | A6 00013 | MOVW | #8, QUOTA_FIB+22 | : 1974 |
| | 5A | A6 | | 08 B0 00015 | CLRQ | -(SP) | |
| | | | | 7E 7C 00019 | CLRQ | -(SP) | |
| | | | | 7E 7C 0001B | CLRL | -(SP) | |
| | | | | 7E D4 0001D | PUSHAB | QFIB_DESC | |
| | | | 00D4 | C6 9F 0001F | CLRQ | -(SPT) | |
| | | | | 7E 7C 00023 | MOVQ | #56, -(SP) | |
| | | | 7E | 38 7D 00025 | MOVZWL | CHANNEL, -(SP) | |
| | | | 7E | C6 3C 00028 | CLRL | -(SP) | |
| | | | FEC8 | 7E D4 0002D | CALLS | #12, SYS\$QIOW | |
| | | | | 0C FB 0002F | RET | | : 1977 |
| | | | 00000000G 00 | 04 00036 1\$: | | | |

; Routine Size: 55 bytes, Routine Base: \$CODE\$ + 09C3

```
: 1999      1978 1 GLOBAL ROUTINE COMMON_IO (EFN,CHAN,FUNC,IOSTS,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)=
: 2000      1979 1
: 2001      1980 1 ++
: 2002      1981 1
: 2003      1982 1 FUNCTIONAL DESCRIPTION:
: 2004      1983 1
: 2005      1984 1     This routine simply executes a $QIOW call with the parameters
: 2006      1985 1     supplied.
: 2007      1986 1
: 2008      1987 1 CALLING SEQUENCE:
: 2009      1988 1     COMMON_IO (EFN,CHAN,FUNC,IOSTS,ASTADR,ASTPRM,P1,P2,P3,P4,P5,P6)
: 2010      1989 1
: 2011      1990 1 INPUT PARAMETERS:
: 2012      1991 1     As to $QIOW
: 2013      1992 1
: 2014      1993 1 IMPLICIT INPUTS:
: 2015      1994 1     NONE
: 2016      1995 1
: 2017      1996 1 OUTPUT PARAMETERS:
: 2018      1997 1     NONE
: 2019      1998 1
: 2020      1999 1 IMPLICIT OUTPUTS:
: 2021      2000 1     NONE
: 2022      2001 1
: 2023      2002 1 ROUTINE VALUE:
: 2024      2003 1     As to $QIOW
: 2025      2004 1
: 2026      2005 1 SIDE EFFECTS:
: 2027      2006 1     As to $QIOW
: 2028      2007 1
: 2029      2008 1 --
: 2030      2009 1
: 2031      2010 2 BEGIN
: 2032      2011 2
: 2033      2012 2 BUILTIN
: 2034      2013 2     AP,
: 2035      2014 2     CALLG;
: 2036      2015 2
: 2037      2016 2 EXTERNAL ROUTINE
: 2038      2017 2     SYSSQIOW      : ADDRESSING_MODE (GENERAL);
: 2039      2018 2
: 2040      2019 2
: 2041      2020 2 ! We simply pass the call and its parameters along to $QIOW.
: 2042      2021 2 !
: 2043      2022 2
: 2044      2023 2 CALLG (.AP, SYSSQIOW)
: 2045      2024 2
: 2046      2025 1 END;                                ! End of routine COMMON_IO
```

00000000G 00

0000 00000
6C FA 00002
04 00009.ENTRY COMMON_IO, Save nothing
CALLG (AP), SYSSQIOW
RET: 1978
: 2023
: 2025

DISKQUOTA
V04-000

15
15-Sep-1984 23:38:38
14-Sep-1984 12:19:46

VAX-11 Bliss-32 V4.0-742
[DISKQ.SRC]DISKQUOTA.B32;1

Page 79
(22)

: Routine Size: 10 bytes, Routine Base: \$CODE\$ + 09FA

: 2047 2026 1
: 2048 2027 1 END
: 2049 2028 0 ELUDOM

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

| Name | Bytes | Attributes |
|--------------|-------|---|
| \$MSG_TEXT | 1804 | NOVEC,NOWRT, RD, NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(0) |
| \$MSG_INDEX | 132 | NOVEC,NOWRT, RD, NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2) |
| \$OWNS | 636 | NOVEC, WRT, RD, NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2) |
| \$SPLITS | 1096 | NOVEC,NOWRT, RD, NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2) |
| _LIB\$KEY0\$ | 30 | NOVEC,NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC,ALIGN(1) |
| _LIB\$STATES | 304 | NOVEC,NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC,ALIGN(1) |
| _LIB\$KEY1\$ | 110 | NOVEC,NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC,ALIGN(1) |
| \$CODE\$ | 2564 | NOVEC,NOWRT, RD, EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2) |

Library Statistics

| File | ----- Total | Symbols Loaded | ----- Percent | Pages Mapped | Processing Time |
|------------------------------------|----------------|-------------------|------------------|-----------------|--------------------|
| _\$255\$DUA28:[SYSLIB]LIB.L32;1 | 18619 | 88 | 0 | 1000 | 00:01.9 |
| _\$255\$DUA28:[SYSLIB]TPAMAC.L32;1 | 42 | 30 | 71 | 14 | 00:00.2 |

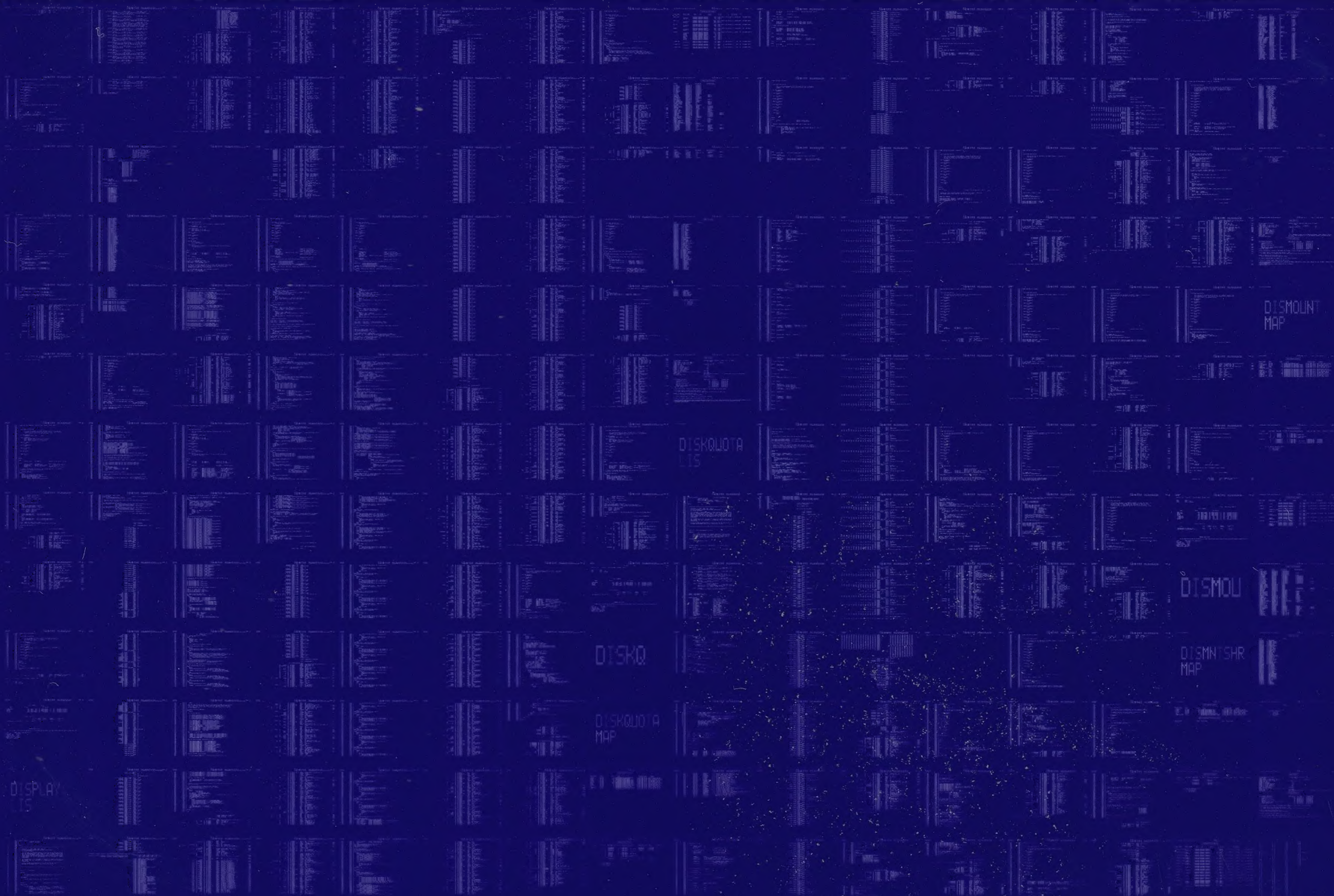
COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:DISKQUOTA/OBJ=OBJ\$:DISKQUOTA MSRC\$:DISKQUOTA/UPDATE=(ENH\$:DISKQUOTA)

: Size: 2564 code + 4112 data bytes
: Run Time: 01:26.1
: Elapsed Time: 02:55.3
: Lines/CPU Min: 1413
: Lexemes/CPU-Min: 57801
: Memory Used: 366 pages
: Compilation Complete

0104 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



DISMOUNT
MAP

DISKJOTA
LIS

DISMOU

DISKQ

DISMNTSHR
MAP

DISKJOTA
MAP

DISPLAY
LIS